ISCTE IUL
Instituto Universitário de Lisboa

DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY

Wearable and IoT Technologies Application for Physical Rehabilitation

Dissertation presented in partial fulfilment of the requirements for the Master's Degree on Telecommunications and Information Science

by

Ricardo José Farinha Alexandre

Supervisors:

Professor Octavian Adrian Postolache, Assistant Professor, ISCTE-IUL

Doutor Vítor Viegas, Assistant Professor, Portuguese Naval Academy

October 2018

Abstract

This research consists in the development an IoT Physical Rehabilitation solution based on wearable devices, combining a set of smart gloves and smart headband for use in natural interactions with a set of VR therapeutic serious games developed on the Unity 3D gaming platform. The system permits to perform training sessions for hands and fingers motor rehabilitation.
Data acquisition is performed by Arduino Nano Microcontroller computation platform with ADC connected to the analog measurement channels materialized by piezo-resistive force sensors and connected to an IMU module via I^2C. Data communication is performed using the Bluetooth wireless communication protocol. The smart headband, designed to be used as a first- person-controller in game scenes, will be responsible for collecting the patient's head rotation value, this parameter will be used as the player's avatar head rotation value, approaching the user and the virtual environment in a semi-immersive way.
The acquired data are stored and processed on a remote server, which will help the physiotherapist to evaluate the patients' performance around the different physical activities during a rehabilitation session, using a Mobile Application developed for the configuration of games and visualization of results.
The use of serious games allows a patient with motor impairments to perform exercises in a highly interactive and non-intrusive way, based on different scenarios of Virtual Reality, contributing to increase the motivation during the rehabilitation process.
The system allows to perform an unlimited number of training sessions, making possible to visualize historical values and compare the results of the different performed sessions, for objective evolution of rehabilitation outcome. Some metrics associated with upper limb exercises were also considered to characterize the patient’s movement during the session.

Keywords: *Internet of Things; Physical Rehabilitation; Pervasive Healthcare; Wearable Sensors; Wearable Wireless Sensor Network; Therapeutic Serious Games; Virtual Reality; Force Sensing; Motion Sensing;*

Resumo

Este trabalho de pesquisa consiste no desenvolvimento de uma solução de Reabilitação Física IoT baseada em dispositivos de vestuário, combinando um conjunto de luvas inteligentes e uma fita-de-cabeça inteligente para utilização em interações naturais com um conjunto de jogos terapêuticos sérios de Realidade Virtual desenvolvidos na plataforma de jogos Unity 3D. O sistema permite realizar sessões de treino para reabilitação motora de mãos e dedos.

A aquisição de dados é realizada pela plataforma de computação Arduino utilizando um Microcontrolador Nano com ADC (Conversor Analógico-Digital) conectado aos canais de medição analógicos materializados por sensores de força piezo-resistivos e a um módulo IMU por I^2C. A comunicação de dados é realizada usando o protocolo de comunicação sem fio Bluetooth. A fita-de-cabeça inteligente, projetada para ser usada como controlador de primeira pessoa nos cenários de jogo, será responsável por coletar o valor de rotação da cabeça do paciente, esse parâmetro será usado como valor de rotação da cabeça do avatar do jogador, aproximando o utilizador e o ambiente virtual de forma semi-imersiva.

Os dados adquiridos são armazenados e processados num servidor remoto, o que ajudará o fisioterapeuta a avaliar o desempenho dos pacientes em diferentes atividades físicas durante uma sessão de reabilitação, utilizando uma Aplicação Móvel desenvolvido para configuração de jogos e visualização de resultados.

A utilização de jogos sérios permite que um paciente com deficiências motoras realize exercícios de forma altamente interativa e não intrusiva, com base em diferentes cenários de Realidade Virtual, contribuindo para aumentar a motivação durante o processo de reabilitação.

O sistema permite realizar um número ilimitado de sessões de treinamento, possibilitando visualizar valores históricos e comparar os resultados das diferentes sessões realizadas, para a evolução objetiva do resultado da reabilitação. Algumas métricas associadas aos exercícios dos membros superiores também foram consideradas para caracterizar o movimento do paciente durante a sessão.

Palavras-chave: *Internet das Coisas; Reabilitação Física; Cuidados de Saúde Perversivos; Sensores Vestíveis; Rede de Sensores Sem Fio Vestível; Jogos Sérios Terapêuticos; Realidade Virtual; Sensores de Força; Deteção de Movimento;*

Acknowledgments

I'm thankful to have Prof. Doctor Octavian Postolache as my supervisor, in this work. Their encouragement, availability and continued support were very important throughout this work and were fundamental to conclude this dissertation. I would also like to thank the Telecommunications Institute (ISCTE-IUL) for access to its installation and material support on project's development. Without forgetting Mr. José Gouveia (Institute of Telecommunications – IST), technical responsible for the electrical assemblies made.

I want to thank my family, in particular to my Mother, for the unconditional support, encouragement and motivation given in recent years, and for all the sewing work done.

Finally, I thank all my friends and colleagues for their help in this work and fellowship during this time.

Thank you all!

This research is supported by Instituto de Telecomunicações, IT-IUL, at ISCTE-IUL, Lisbon, Portugal and Fundação para Ciência e Tecnologia: project PTDC/DTP-DES/6776/2014.

Content

List of Figures

List of Tables

List of Acronyms

ADC	Analog-to-Digital-Converter
AHRS	Attitude and Heading Reference System
API	Application Programming Interface
COM	Component Object Model
DB	Database
DBMS	Database Management System
FSR	Force Sensitive Resistor
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
I^2C	Inter-Integrated Circuit
IMU	Inertial Measurement Unit
IoT	Internet of Things
MEMS	Micro Electro Mechanical Systems
OS	Operating System
PCB	Printed Circuit Board
REST	Representational State Transfer
VR	Virtual Reality
UWP	Universal Windows Platform

Chapter 1 - Introduction

1.1 Motivation and Overview

The health sciences are areas of study related to life and include several branches such as medicine, biomedicine, nursing, speech therapy, clinical criticism, pharmacy, physical health, dentistry, psychology, occupational therapy, nutrition, physiotherapy. In all these fields the use of science, technology, engineering or mathematics to provides health care to human beings [1] is considered.

The health sciences are divided into three methodological and action principles [2]:

- *In the diagnostic phase*: it includes anamnesis, probing, clinical exams, laboratory exams and tests.
- *In the performance phase*: it is defined by indication of medications, application of manoeuvres, massages, specific therapeutic exercises and physical activities, diet guidelines, postures and behavioural change, among others.
- *In the follow-up phase*: through comparisons of exams and tests, clinical evaluation and periodic returns.

In the last decades, the main sociodemographic trends are defined by the increase in the percentage of population living in cities, as well as by the aging of the population itself. According to some studies, by 2050, the number of older people on the planet will exceed that of young people for the first time in human history [3], and about 70% of the world's population must live in urban areas [4]. These trends make it necessary to integrate new technologies and innovative solutions that respond to new environmental, social and health challenges in order to offer new services that can optimize the quality of life of the cities.

In societies many people experiencing physical and / or motor limitations associated with a variety of reasons, whether due to birth problems, work-related accidents or some of the restrictions caused by aging. Elderly people are more affected by motor impairments that made difficult to perform even simple daily tasks, such as picking up an object without difficulty, eating alone, or even dressing. Such consequences may restrict personal activities and avoid the fully participating of elderly in the community that conducts to negative effect on their daily work and life in general.

The use of physical therapy methods performed in specialized clinics are very important to help people recover partially or fully their physical and/or motor limitations. Performing the physiotherapy exercises tailored to their needs made possible to reduce the existing physical limitations that assures an active and healthy life.

The Internet of Things (IoT) as 21st century technology presents a great potential to change and replace various methods of classical medicine to improve the wellbeing of humans. Within the area of medical sciences, several studies focus on bringing this technology closer to the users integrating it in the process of physical rehabilitation. On the other hand, the health sector remains at the top, among the fastest to adopt Internet of Things. The application of the IoT in the healthcare area covers areas such physiological monitoring, autonomy assistance and human comfort solutions.

This tendency to make medical device IoT compatible may contribute to the quality and effectiveness of the healthcare services, bringing especially high value to the elderly and to the patients with chronic diseases that requires constant monitoring of their health conditions. According to some estimates [5], spending on the Healthcare IoT solutions will reach a staggering $1 trillion by 2025 and, hopefully, will set the stage for highly personalized, accessible, and on-time Healthcare services for everyone. The tendency to assure IoT compatibility of the medical devices improves the quality and effectiveness of the service, assuring continuous healthcare monitoring of elderly, patients and patients with chronic disease.

The advantages of performing physical parameter measurements using equipment connected to Internet by comparison with the conventional ones are: the storage of data in a remote database accessible by Internet; the data analyse of measurements; the perform of specific actions based on the results of these measurements.

An objective evaluation based on the measurement of motor activity parameters, especially the functionality of the upper and lower limbs during the rehabilitation period, offers the possibility to increase patients' motivation through adequate feedback and also provides data associated with the physical rehabilitation reports that can be used as the starting point for improved communication between physiotherapists and rehabilitation procedures implemented, but also to improve communication between the physiotherapist and the user of rehabilitation services. Good therapeutic communication may imply a greater ability to obtain valid informed consent, positive clinical outcomes, higher levels of patient satisfaction, higher levels of patients' compliance with rehabilitation programs, lower levels of patient frustration [6].

In addition, the health community has shown a great interest in therapeutic approaches based on computer therapeutic serious games. The concept of therapeutic serious games refers to the use of computer games without the primary purpose of providing pure entertainment [7]. Virtual reality (VR) implementation for training exercises in physiotherapy interventions has shown important advantages both as technique for objective assessment of treatment as well as for engagement of patient in the healthcare process [8]. According to the experience described [9], the integration of VR technology with therapeutic serious games provides increased motivation during the rehabilitation process.

1.2 Objectives and Research Questions

The purpose of this research is design and development a system composed by hardware and software tools. The hardware supports the signal acquisition and primary processing of data. It combines a set of wearable devices to be used in natural interactions with a set of highly interactive serious therapeutic serious games developed in a virtual reality gaming platform. The game scenario provides elements associated with rehabilitation exercises for user hands and fingers.

This system aims to introduce metrics to evaluate the clinical status of the patient and improve his physical limitations, complementing traditional rehabilitation systems in the physiotherapy area. The use of serious games in VR scenarios in conjunction with intelligent clothing devices allows the patient with physical and/or motor limitations the possibility of performing traditional exercises in a highly interactive and non-intrusive way, aiming at motivational improvements, allowing different activities to be performed repeatedly over a pre-set period.

By integrating sensing and VR technologies, it is possible to create a gaming platform for the patient that can be used during their rehabilitation. Regarding the physiotherapist's point of view, it is possible to configure parameters related to rehabilitation plan that actually will be expressed by serious games settings adapted the specific needs of each patient. The obtained results from the use of the games are stored on a remote server, which will help the physiotherapist to evaluate the performance of the patients in different physical activities during the rehabilitation sessions using a mobile device. Thus, Mobile Application was developed for configuration of the games and visualization of the results. The system allows the execution of an unlimited number of training sessions. In this way, it is possible to visualize historical values and compare the results of the different training sessions, analysing their evolution.

This research work aims to obtain an answer to certain questions, such as:

1. What are the advantages of creating a platform capable of making measurements with intelligent sensor devices applied to clothing?
2. What are the advantages of storing the results of measurements made with smart sensor devices applied to clothing?
3. Will it be possible to develop metrics that can help physiotherapists assess the patient's progress during the motor rehabilitation period?
4. What will be the performance of the wearable system in finger motion detection?
5. It is possible to create highly configurable and highly interactive serious games which will be part of the process of rehabilitation and motivation of the patient?
6. What are the advantages of wearable sensing and VR as part of new method for rehabilitation?
7. What is the acceptance of the system by the patients and by the physiotherapists?

1.3 Research and Planning Methods

The research method used in this project was based on the Agile Methodology, but due to its being composed of software and hardware, some changes were considered. The Research Method Flow consists of five distinct phases and is presented below (see Figure 1.1):

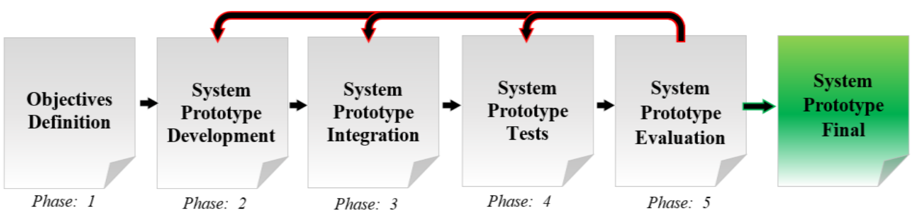

Figure 1.1 - Research Method’s Flow.

Phase 1: *Objectives Definition* - In this phase, a set of technologies, processes, tools and methodologies inherent to physical-motor rehabilitation are studied, to know their impacts on the rehabilitation of patients in the physiotherapy area. These studies serve as guidelines for research, to which the project must obey;

Phase 2: *System Prototype Development* - In this phase the different segments and components that will be part of the project are developed to check with the established objectives. The objectives are:

- Design of the Physio-Wearable system architecture;
- Design of the electrical circuit of the components to be used, refers to the circuits applied to the gloves and to the headband, with microcontroller performing primary processing of the data collected from intelligent sensors;
- Development of the serious games applied to physiotherapy of hands and fingers with support of the developed circuits, using Unity 3D Engine;
- Creation and structuring of a database for remote storage of all data generated by the system:
- Development of the application for mobile devices, developed for Cross-Platform (Android + iOS + UWP) and intended for physiotherapists and patients, serving as support to see information and results collected;

Phase 3: *System Prototype Integration* - In a sequential structure, the different components developed in the previous phase must be integrated to create a prototype of the system;

Phase 4: *Tests of the system prototype components* - A set of tests and adjustments in the system components are performed to obtain a final prototype;

Phase 5: *Evaluation of the system prototype* - The final prototype of the project must be tested and evaluated.

The phases 2, 3 and 4 are cyclical stages. At the end of the fifth phase in case the final prototype is in accordance with the objectives defined in the first phase, the project is finalized. Otherwise, it is necessary to analyze the problem or improvements to be made and return to the research phase necessary to combat the problems between the initially defined objectives and the final prototype.

1.4 Structure of the Thesis

The thesis is composed of 8 chapters: Chapter 2 includes a literature review on Serious Therapeutic Games, Smart Sensors, Smarts Controllers and Mobile Applications Software. Chapter 3 describes the entire Physio Wearable system and how it works, as well the Embedded System. The serious games applied to physical rehabilitation are explored in Chapter 4. Chapter 5 describes the implementation of an information system for storing data, mainly expressed by database. Chapter 6 describes the support for physical therapists to create training plans for the various patients and the ability to visualize their results using a mobile application. Chapter 7 presents the tests, the obtained results and evaluation performed. Chapter 8 presents conclusions and future work. A user manual of the system applications and a technical manual explaining the main functions of the applications was included in the thesis.

Chapter 2 - State of the Art

Overview

This chapter presents the scientific literature research regarding serious therapeutic games, their application and use of intelligent sensors. In addition, some commercial medical software applications existing for this purpose were analysed.

2.1 Serious Games and Virtual Reality

A **Serious Game** is software or hardware developed through the principles of interactive game design, with the goal of transmitting educational or training content to the user. The term "serious" refers to that the game is designed more for educational purposes than for entertainment purposes [10]. They have been widely used in the areas of defence, education, scientific exploration, health services, emergency management, urban planning, engineering, religion and politics in an immersive or interactive way that can be enjoyed in the best possible way [11]. Some examples below:

▪ ***Food Force*** is an educational game published by the United Nations World Food Programme (WFP) in 2005. In 2011 has a new release in partnership with Konami and Facebook [12]. Players take on missions to distribute food in a famine-affected country and to help it to recover and become self-sufficient again. At the same time, they learn about hunger in the real world and the WFP's work to prevent it (see Figure 2.1).

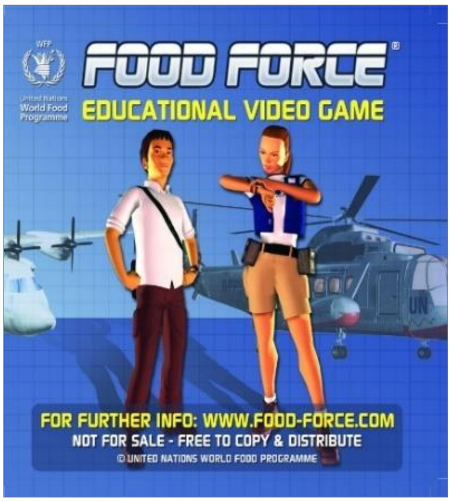

Figure 2.1 - Food Force Serious Game [12].

▪ ***vHealthCare™*** (BreakAway Games) is a first-person, immersive virtual environment used to train healthcare professionals, such as nurses, doctors, first responders, in clinical diagnostic skills and decision-making protocol and to teach professional how to respond to catastrophic incidents, such as combat, bioterrorism, or major incidents [13] (see Figure 2.2).

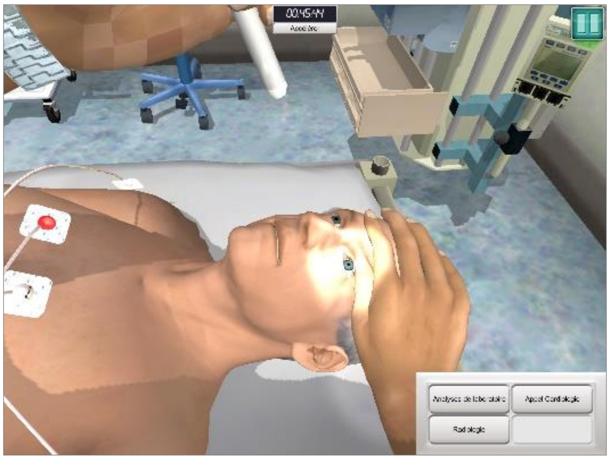

Figure 2.2 - vHealthCare Serious Game [13].

▪ ***Mursion*** [14] is the virtual reality environment where professionals practice and master the complex interpersonal skills necessary to be effective in high-stress professions. Used by a wide range of innovative organizations in Healthcare, Education, Hospitality, Finance, and other corporate sectors, delivers customized virtual reality training simulations that recreate the most demanding interpersonal challenges that professionals confront on the job every day. While the media has focused on virtual reality as a tool for gaming and other forms of entertainment, virtual reality simulations are on the verge of revolutionizing professional training (see Figure 2.3).

Figure 2.3 - Mursion Virtual Reality Environment [14].

▪ **MIRA Rehab** [15] is a rehabilitation support solution that uses motion-tracking sensors to gamify physical therapy and increase patient compliance. The solution is a software platform that transforms physical and cognitive exercises into clinical exergames, making therapy more convenient and easier to follow, while evaluating and reporting patient compliance. MIRA has been used successfully in orthopaedic and neurological therapy for children and adults, as well as in fall prevention and active aging programs for the elderly (see Figure 2.4). This system contains over 450+ exergames, along with assessment tools, cognigames and custom exercises. The content is clinically built based on input and feedback from therapists and patients and is continuously expanded and improved through period application updates.

Figure 2.4 - MIRA Rehab applications [15].

In particular, in serious games, the goal is related to training and learning. It is intended to convey messages for training purposes or to develop certain skills and behaviours that are transferable to the real world. All these purposes can be accomplished using artificially generated conflicts to create challenges for users, according to a principle of deliberate practice [16], that is, "presents interpreters with tasks that are initially out of their current level of reliable performance, but can be mastered with practice, gradually refining performance through repetitions after feedback." The main requirements of deliberate practice are demanding (but not impossible) tasks, specific goals for improvement, continuous feedback and opportunities for repetition. In this way the serious Games can offer the right environment to fulfil the stated requirements.

There are many types of games, they can come in a variety of formats, ranging from pen-and-paper-based games or board games, to more advanced games based on computer technology (i.e. simulation) and advanced graphics capabilities. In our research we will explore the exergames (concept of games that help people to exercise) [17]. In fact, using games instead of (or sometimes more appropriately, in conjunction with) traditional methods has been proven to be more effective when the outcome to be achieved is not merely notional. Research literature [18] suggests that passive processes are less effective than interactive and engaging, regardless of target audience. Games can arouse emotions (fun, happiness, anger, frustration, etc.) that seem to open channels directly into our minds, allowing for a deeper understanding of new knowledge, concepts and abilities, and a longer retention in our memory. In addition, to succeed or to continue in the game, players usually need to develop a variety of strategies and skills by applying the new knowledge gained in the previous game held, so the games are educationally correct as they require the player the ability to learn and remember rules, game mechanics and processes.

In a study presented [19], the authors analyze the impact of serious games in terms of prevention and rehabilitation processes. The authors describe the benefit of serious games by applying the following criteria:

- *Efficiency and effectiveness of the intervention.* If Serious Games are not effective and efficient in improving the physical, psychic, sensory and / or social functions of people with motor difficulties, they cannot be considered a reasonable means to improve health or well-being.
- *Additional benefits, such as psychological factors.* In addition to effectiveness and efficiency, a Serious Game should offer additional advantages like fun, enjoyment and adherence.
- *Quality of the study.* The study should consider the relevant aspects of the game's effects, including psychological, social, environmental and behavioural factors.

The study concludes that serious games produce substantial benefits for a portion of patients who are using this kind of technology, however the games must be developed to be completely adaptable to the needs of the patient, and it is necessary to evaluate some configurations to support motivation through different systems social policies. Emotional and psychological impact of the game can be immense.

Virtual Reality scenarios have also been used to support the socio-emotional development of adolescents with autism spectrum disorders and to treat different types of psychological phobias, spider phobia, social phobia / public speaking anxiety, fear of driving / flying, agoraphobia, and other mental health problems.

Several studies [20] [21] suggest stroke as the most significant cause of disability in adults. Medical reports showed that six months after stroke, 49% of patients needed help with bathing, 31% of patients needed help with dressing, and 33% of patients needed help with feeding. Research suggests that intensive and repetitive training may be necessary to modify the neural organization. Stroke rehabilitation is one of the most successful health research areas where Virtual Reality technologies and games have been applied to create new intervention tools and have influenced our current practice in post-stroke physical therapy.

Video games are not just about providing pure fun or entertainment for the gamer, they can be used to help people to improve their physical and / or motor skills. Virtual Reality environments are increasingly used through games in various areas of the medical sciences also in physiotherapy. This new generation of tools available for use in physiotherapy and rehabilitation has seen an exponential growth over the last few years. Through this resource it is possible to perform a rehabilitation treatment in a more fun and perhaps motivating the patient to carry out with more commitment the proposed exercises. One of the essential features of virtual reality focuses on Dynamic Simulation, also based on physical simulation.

Dynamic Simulation creates realistic movements of virtual objects based on Newton's Laws of Motion. The behaviour of virtual objects and their responses to external force and torque are simulated in a physically realistic manner. Physics simulation models' objects with physical properties such as mass, inertia, barycentre, joint constraints, restitution, and surface friction - can make objects in virtual worlds look not real but real. Typical dynamic simulation includes collision detection (and response) and simulation of gravity, frictional force and kinematics in motor actions. Physics can be applied to rigid bodies or deformable bodies, such as human tissues. Rigid body dynamics play an important role in motor rehabilitation and is especially necessary when it comes to the skilful manipulation of virtual objects.

The advantage of VR usage in clinical motor rehabilitation environments is related to the possibility to program the VR scenario according to type and difficulty of the tasks associated with the therapeutic plans. The Dynamic Simulation offers more flexibility in experimental setup, in which not only positions, orientations and object sizes can be modified, but also gravity, restitution, force and torque dampers, and joint constraints can be modified safely, thus creating a set individualized rehabilitation tasks, which would be something impossible to achieve with the user's residual motor control in the physical world. These tasks can be reliably simulated over a period of several weeks or months, and the outcome of the participant can be evaluated.

The addition of dynamic simulation to VR therapy can increase patients' immersion experience in virtual environments, which in turn can increase motivation and activity utilization and thus improve clinical outcome. The quality of the dynamic simulation required in a Virtual Reality system is application specific. In the cognitive and affective domains of learning, where the focus of training is more on attitudes, such as using VR therapy to treat phobia, high physical fidelity is not always necessary [22].

In serious gaming for health care, dynamic simulation can be implemented using next generation gaming engines, either commercial engines or open source engines. Notably, many commercial game engines, such as (see Figure 2.5):

- NVIDIA PhysX [23];
- Epic's Unreal Engine [24];
- Unity Technologies' Engine [25],

Figure 2.5 - Example of many commercial game engines.

provides free licenses for educational and non-commercial applications or for independent developers (for example, stand-alone games). These tools have been adopted in a prosperous way in the indie gaming industry recently, and also provide great low-cost solutions for serious gaming developers.

Video games such as Nintendo Wii Sports (boxing, tennis, bowling and golf etc.) [26] have been used for rehabilitation of patients with cerebral palsy and hemiplegia to aid recovery and recovery of strength and ability through repetitive physical exercises [27] (see Figure 2.6).

Figure 2.6 - Example of using Wii Sports in physical rehabilitation process.

The intuitive control design of the Wii enables control through natural movements without restricting sensors and cables. Wii Sports and Wii Fit are currently being used in rehabilitation therapy for patients undergoing treatment after stroke, cranioencephalic trauma, spinal cord injury and combat injury. Previous research [28] has shown that active Wii gameplay is an effective tool for exercise, and that Wii games are more useful in limb extension and calorie burning exercises. Hines Veterans Hospital in Chicago even installed a Wii gaming station in its spiral wound unit to aid in physical rehabilitation after several surgical procedures [22].

2.2 Smart Sensors

The smart sensors have been identified as an effective and efficient solution for the rehabilitation area because of its potential in monitoring of patient movement. We can divide the sensors into two categories:

- ***Wearable*** - Sensors that need to be connected to some part of the human body;
- ***Non-wearable*** - Sensors that do not need to be attached to any part of the body, being placed somewhere so that it is possible to identify patient-related movements and collect data.

In a study [29], researchers note that the development of technologies in recent years has facilitated the deployment of equipment-level sensors for clinical applications, including rehabilitation. The interest of researchers and clinicians in sensor-attached equipment has led to a shift in the field of usable technology where new developments and designs of sensors and systems are highlighted. Advances in the field of microelectronics allow the development of small circuits and the possibility of adding many sensors, such as Electromyography (EMG) sensor rate, Electrocardiography (ECG) measurement, gyroscopes or accelerometers (see Figure 2.7).

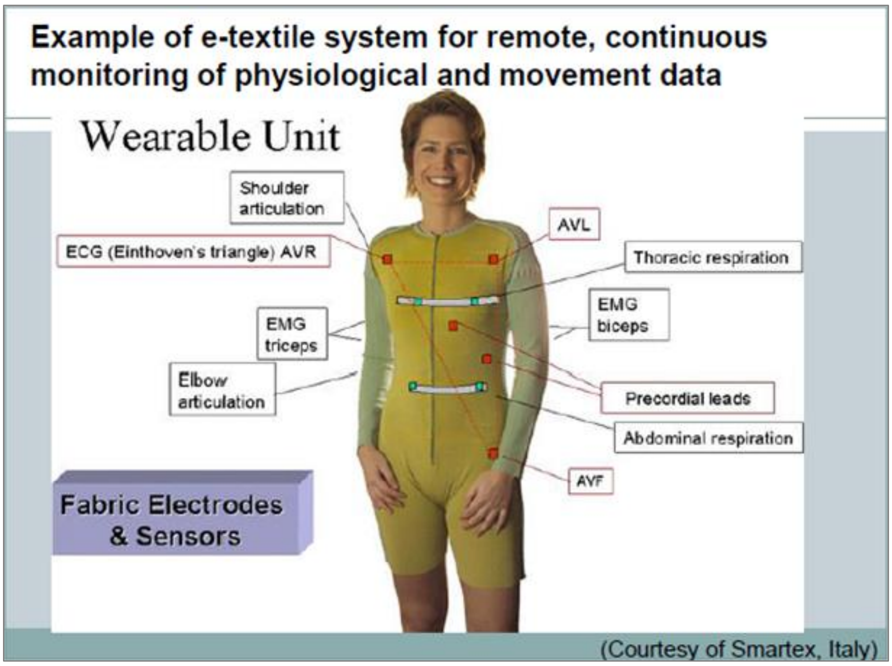

Figure 2.7 - Example of e-textile system for remote, continuous monitoring of physiological and movement data [29].

The fun that the games propose combined with sensing technologies, make possible the development of new training systems for physical therapy, based on psychological motivation of patients and on extension of the training time in performing exercises repeatedly with high quality of the motion. The disadvantage of some motion sensors is that users need to attach them to body limbs or hold them in their hands to detect movements, which can cause discomfort and inconvenience to those who use them. Strategies involving the use of various technologies for people with motor disabilities have been developed for rehabilitation through numerous implementation, VR and motion-based serious games have been used recently for rehabilitation. The type of exercises can be from simple goal-guided movements to learning the execution of daily tasks.

2.2.1 Smart Controllers

Some examples of using game platforms with smart motion controllers are:

- **Nintendo Wii Remote** (remote device built-in accelerometer offering position data (X, Y, Z) and a high-resolution infrared sensor in which it provides data on rotations (Yaw, Pitch, Roll)) to interact with Nintendo Wii Console (see Figure 2.8).

Figure 2.8 - Nintendo Wii Remote**.**

- **PlayStation Move** [30] (remote device Sony PlayStation, involves the use of a Play Station Eye (video-camera)) to interact with Play Station Console (see Figure 2.9).

Figure 2.9 - PlayStation Move.

- **Microsoft Kinect** [31] - A video game interface camera with a detection of the human body. It detects changes in infrared light patterns, the coordinates of the body joints being detected. The spatial resolution does not allow measuring small segment movements, such as fingers. This system frees the player from any controllers to interact with Xbox Console (see Figure 2.10) A new version of Kinect V2 was release later to the market.

Figure 2.10 - Microsoft Kinect V1 and KinectV2.

▪ **Leap Motion Controller** [32] – Provides high resolution in detecting small movements, like rotations and velocities. It has been specially designed for hand and finger movements (see Figure 2.11).

Figure 2.11 - Leap Motion Controller.

▪ More recently the great trend of the gaming market already integrates sets of VR goggles or Head-Up-Display (HUD), where the immersion of use is much superior. Example of this is the **PlayStation VR** [33] (see Figure 2.12) or the **HTC Vive** [34] (see Figure 2.13) (these combines a Head-Up-Display (HUD) with remote device controllers).

Figure 2.12 - PlayStation VR.

Figure 2.13 - HTC Vive.

In [35], authors present a system developed for physical rehabilitation that can be performed in clinics or at home, allowing the physiotherapist to monitor the progress of physical rehabilitation of patients. The system is divided into three components: a video game developed in Unity3D, based on real-life conditions, along with virtual reality, supported by the Microsoft Kinect V2 sensor for data collection, an API (Application Programming Interface) and a mobile application for Android OS devices for viewing the data. *Apples Harvesting* (see Figure 2.14) and *Therasoup* (see Figure 2.15) are 3D Games present on system that can be adapted to the patient's clinical needs.

Figure 2.15 - *Apples Harvesting* Serious Game developed in Unity 3D [35].

Figure 2.14 - *Therasoup* Serious Game developed in Unity 3D [35].

In [36], authors present another system developed for physical rehabilitation that can be performed in clinics or at home, allowing the physiotherapist to monitor the progress of physical rehabilitation of patients based on finger movements. The system is divided into three components: a set of video games developed in Unity3D (see Figure 2.16), based on real-life conditions, along with virtual reality, supported by the Leap Motion Controller for data collection, an API (Application Programming Interface) and a mobile application for Android OS devices for viewing the data.

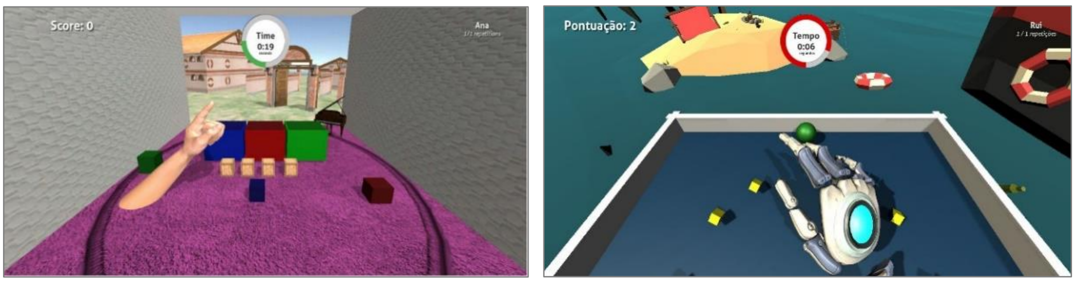

Figure 2.16 - CollectCube and PickaBall Serious Games developed in Unity 3D gaming platform to LeaPhysio System [36].

2.2.2 Inertial Sensors

All body movements are characterized by concurrent accelerations and decelerations, which can be applied for tracking or motion detection. Micro-machined inertial sensors, including accelerometers and gyroscopes, are silicon-based sensors. They are small in size and can be worn on the body. The principle of operation of these sensors is based on the measurement of inertia and can be applied anywhere without reference.

With recent technological advances, the measurement sensors have become much smaller. The development of Micro-Electro-Mechanical-Systems (MEMS) allowed sensors to generate data faster, operate on batteries, communicate wirelessly and provide easier wearability. The sensors can be divided into two main categories: physiological and mechanical and have a wide variety of applications. Our interest in this work lies in three different mechanical sensors that are accelerometers, gyroscopes and magnetometers.

Accelerometers are instruments that measure applied acceleration acting together in a sensitive axis. A single-axis accelerometer consists of a mass, suspended by a spring in the housing. The mass may move in one direction and the mass displacement is a measure of the acceleration and gravity difference along the sensitive axis given by the unit vector. To track human movement, one is needed for each of the three axes of motion (x, y and z). A triaxial accelerometer (TA) can be built by mounting three of these single axis accelerometers together or using a single mass with three degrees of translational freedom. They can provide information about their angle of inclination relative to the descending direction, detecting the acceleration due to gravity.
However, these sensors are unable to distinguish between gravitational force and actual acceleration. The readings are severely affected by noise. The sensor converts the acceleration into a voltage, which is later translated into a binary number that an autopilot can understand. There are different types of accelerometers, though the most common are Capacitive and Piezoelectric, which basically measure voltage variations due to the sensor's deformation.

Gyroscopes are used to detect angular velocities. However, they are unable to determine their absolute orientation and suffer from deviation problems. The drift adds small angular velocities even when the device is completely stopped. Over time, these accumulate and affect the overall sensor reading. Gyroscopes measure angular velocity and can be combined with accelerometers to estimate orientation. A combination of accelerometers and gyroscopes will then be an approach to obtain both inclination and orientation information. Typically, angular orientation of a body segment is determined by integrating the output from the angular rate sensors strapped onto the segment. A relatively small offset error on the gyroscope signal will introduce large integration errors.

The magnetometer is an instrument for measuring the direction and/or intensity of magnetic fields. It is sensitive to the earth's magnetic field. It gives information about the heading direction in order to correct drift of the gyroscope about the vertical axis. They use the Earth's field as a reference direction. Since the Earth's magnetic field is quite weak, it is important to be careful with the use of the magnetometer close to other magnetic fields and to keep the metallic objects close to avoid magnetic interference [37].

Measurement of human movement can be summarized into three types: inclination (angle to the vertical), orientation (relative angle of limbs or body) and daily life monitoring (for example, monitoring activity, training of motor skills), the latter requires both measurement of inclination and orientation.

The use of accelerometers, gyroscopes and magnetometers together can produce better measurements than possible with any of these sensors individually. This combination produces more accurate information about motion using a sensor fusion algorithm. The device is often referred to collectively as the **Inertial Measurement Unit (IMU)** [38].

2.2.3 Glove-Based Tracking Devices

Since the late 1970s, people have studied glove-based devices for the analysis of manual gestures. Glove-based devices adopt sensors attached to a glove that transduces flexion and abduction of the finger into electrical signals to determine hand posture (Zhou, 2004) [39].

In 1977 Daniel J. Sandin and Thomas Defanti at the Electronic Visualization Laboratory, a cross-disciplinary research lab at the University of Illinois at Chicago, created the *Sayre Glove*, the first wired glove or data glove, it was based on an idea of a colleague at the laboratory, Richard Sayre. An inexpensive, lightweight glove to monitor hand movements, the *Sayre Glove* provided an effective method for multidimensional control, such as mimicking a set of sliders [40].

The *Dataglove* (originally developed by Thomas Zimmerman in 1982 on VPL Research [41], the pioneer of virtual reality technology and 3D network graphics) is a neoprene fabric glove with two fiber-optic loops on each finger. Each loop is dedicated to a joint and at one end of each loop is an LED and at the other end is a photosensor (see Figure 2.17). The fiber-optic cable has small cuts along its length. When the user bends a finger, the light escapes from the fiber-optic cable through these cuts. The amount of light reaching the photosensor is measured and converted to a measure of how much the finger is bent. *Dataglove* requires recalibration for each user.

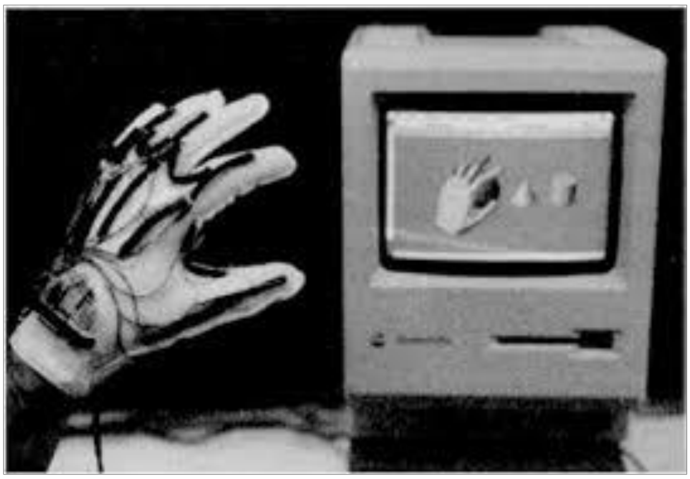

Figure 2.17 - The Z™ Glove developed by Zimmerman.

The first glove to use multiple sensors was offered by the "Digital Entry Data Glove" which was developed by Gary Grimes in 1983. It used different sensors mounted on a cloth. It consisted of touch or proximity sensors for determining whether the user's thumb was touching another part of the hand or fingers and four "knuckle-bend sensors" for measuring flexion of the joints in the thumb, index, and little finger. It also had two tilt sensors for measuring the tilt of the hand in the horizontal plane and two inertial sensors for measuring the twisting of the forearm and the flexing of the wrist. This glove was intended for creating "alphanumeric characters" from hand positions.

One of the first wired gloves available to home users was the *Nintendo PowerGlove* [42], in 1987. This was designed as a gaming glove for the Nintendo Entertainment System. It had a crude tracker and finger bend sensors, plus buttons on the back (see Figure 2.18). The *Nintendo PowerGlove* was the first to really capture people's imagination. It wasn't very well supported, and had its share of technical problems, but its futuristic design and intuitive interface inspired writers and cyberpunks for a generation. Data gloves appeared in several sci-fi films like Johnny Mnemonic and The Minority Report, and inspired the gesture-based computers seen in Iron Man.

Figure 2.18 - The *Nintendo PowerGlove* developed by Zimmerman [42].

Based on design of *Dataglove*, *PowerGlove* was developed by Abrams-Gentile Entertainment (AGE Inc.) for Mattel through a licensing agreement with VPL Research. *PowerGlove* consists of a sturdy *Lycra* glove with flat plastic strain gauge fibers coated with conductive ink running up each finger; measures change in resistance during bending to measure the degree of flex for the finger as a whole. It employs ultrasonic system (back of glove) to track roll of hand (reported in one of twelve possible roll positions), ultrasonic transmitters must be oriented toward the microphones to get accurate reading; pitching or yawing hand changes orientation of transmitters and signal would be lost by the microphones; poor tracking mechanism. Whereas the *Dataglove* can detect yaw, pitch and roll, the *PowerGlove* can only detect roll, and provides 4D information, i.e. x, y, z and roll.

This was followed by the *CyberGlove*, created by Virtual Technologies, Inc. in 1990. Virtual Technologies was acquired by Immersion Corporation in September 2000. In 2009, the *CyberGlove* line of products was divested by Immersion Corporation and a new company, *CyberGlove Systems LLC*, took over development, manufacturing and sales of the *CyberGlove*. In addition, Immersion Corp also developed three other data glove products: the *CyberTouch*, which vibrates each individual finger of the glove when a finger touches an object in virtual reality; the *CyberGrasp* which actually simulates squeezing and touching of solid as well as spongy objects; and the *CyberForce* device which does all of the above and also measures the precise motion of the user's entire arm (see Figure 2.19).

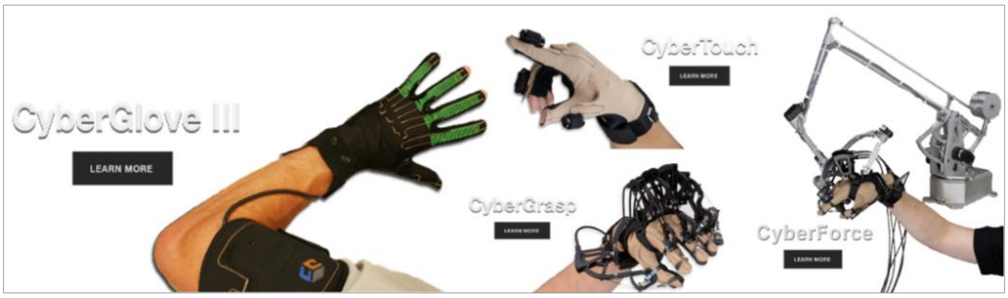

Figure 2.19 - *CyberGlove* Products.

The CyberGlove System [43] consists of a *CyberGlove*, its instrumentation unit, a serial cable to connect to a host computer, and an executable version of the software and calibration software of the VirtualHand manual graphic model. CyberGlove has a software programmable switch and LED on the bracelet to allow the system software developer to provide the *CyberGlove* user with additional input / output capability. The instrumentation unit provided a variety of convenient functions and features, including captures the movement of a user's fingers and hand, and, in conjunction with the software, maps the movement to a graphical hand on the computer screen, allowing users to “reach in and manipulate” digital objects as if they were physical objects.

In 2002, the P5 Glove was released that it worked like a two-dimensional mouse and some computer games were specially adapted to provide "3D" support for it. The P5 glove is compatible with Microsoft Windows XP and the classic Mac OS. Unofficial drivers for Linux also exist. Although it received some positive reviews of gadgets and game magazines, its lack of compatible software and other problems made it a novelty. Since then, it has been discontinued (see Figure 2.20).

Figure 2.20 - The P5 Glove.

Following the P5 Glove it was released the 5th Glove. A data glove and flexor strip kit sold by Fifth Dimension Technologies [44]. The package uses flexible optical-bending sensing to track hand and arm movement. The glove can be used with 5DT's ultrasonic tracking system, the 5DT Head and 5DT Hand tracker, which can track movement from up to two metres away from the unit's transmitter (see Figure 2.21).

The 5DT Data Glove Ultra is designed to satisfy the stringent requirements of modern Motion Capture and Animation Professionals. It offers comfort, ease of use, a small form factor and multiple application drivers. The high data quality, low cross-correlation and high data rate make it ideal for realistic Realtime animation.

Figure 2.21 - 5DT Data Glove [44].

The system interfaces with the computer via a USB cable. A Serial Port (RS 232 – platform independent) option is available through the 5DT Data Glove Ultra Serial Interface Kit. It features 10-bit flexure resolution, extreme comfort, low drift and an open architecture. The 5DT Data Glove Ultra Wireless Kit interfaces with the computer via Bluetooth technology (up to 20m distance) for high speed connectivity for up to 8 hours on a single battery. Right and left-handed models are available. One size fits many (stretch *Lycra*) [44].

2.2.4 Motion Tracking

Human motion tracking systems generate real-time data representation of human movement. Sensors or markers are placed on the body to measure the distance, or orientation or position of an external source, like light sources or cameras. Alternatively, sensors or markers can be attached to anatomical landmarks in the body and used to calculate their relative positions in space.

Motion tracking systems can be classified according to the position of the sensors and sources, or according to motion tracking techniques, such as orientation and electromagnetic positioning trackers, position and acoustic guidance trackers, electrostatic positioning, and trackers. orientation, and video and electro-optical tracking systems, for example.

Zhou et al (2004) [39] have suggested the following classifications: non-vision-based system, marker-based tracking systems with marker-based tracking systems and robot-guided tracking systems. Systems such as Qualisys [45], VICON [46] and CODA [47] are marking-based tracking systems.

Qualisys is a provider of precision motion capture and 3D positioning tracking systems for engineering, biomechanics, animation, virtual reality, robotics, and movement sciences. Their optical tracking technology, also known as optical motion capture, makes it possible to measure the position of very fast-moving objects with extremely high accuracy. Indoor, outdoor, ground-to-air or underwater. The technology consists of motion capture cameras, software and other hardware, designed and produced by company. Their technology is used by researchers, bio mechanists, physicians and engineers from all over the world.

For example, the motion capture systems can be used to find and prevent injuries, improve performance in biomechanics and help engineers build better products and increase production efficiency. The Figure 2.22 shows an example of integration Qualisys System in gait rehabilitation.

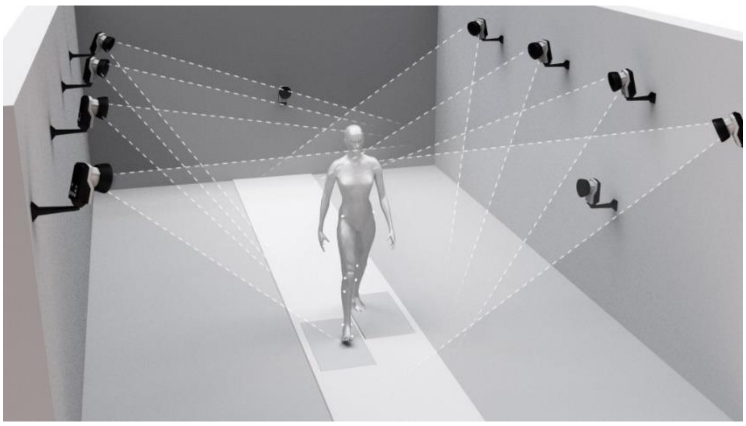

Figure 2.22 - Example of integration of Qualisys System in gait rehabilitation [45].

The Vicon System [46] (not used in video games) capable of performing motion capture for the science, entertainment and engineering industries. The Vicon Biomechanics and Sports Science community encompasses many applications including research, sports performance and animal science. Motion analysis for improving performance or injury prevention, whether clinical, research or educational, requires a fully flexible motion measurement system capable of capturing movement in all environments with minimal configuration (see Figure 2.23).

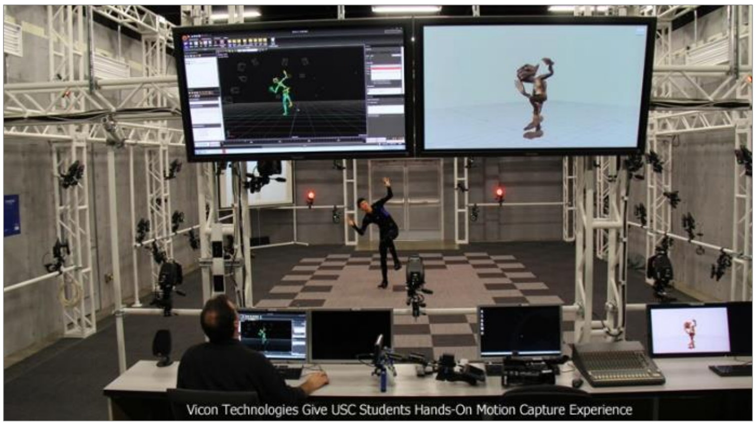

Figure 2.23 - Example of using VICON System [46].

CODA is a supplier of motion capture equipment to the academic research, clinical and other related life science markets, with the goal of enhancing lives (see Figure 2.24).

Figure 2.24 - Example of using CODAmotion System [47].

The Xsens is the innovative leader in 3D motion tracking technology [48]. Its sensor fusion technologies enable seamless interaction between the physical world and the digital world in consumer devices and professional applications such as 3D character animation, motion analysis, and industrial stabilization and control. They apply AHRS (Attitude and Heading Reference System) algorithms to merge various inertial sensors (see Figure 2.25). Xsens' products are applied in different fields, such as gaming production, film production, sport performance analysis, among others, and not use cameras.

Figure 2.25 - Example of using Xsens System integrated with Unity 3D [48].

All commercial systems available above require specialized configuration, dedicated space and expensive equipment. They also restrict the movement of the body to some degree. For example, robot-guided tracking systems are limited in the range of human movement and, in particular, in walking movements or other transitional movements. Glove-based systems are hampered by wires connected to each sensor. 3D optical tracking systems, such as Vicon or CODA, are being used for rehabilitation in clinical settings. They comprise one or more video cameras or sensor arrays, PC interface and active or passive markers.

2.3 Information Storage

In the health system, due to the registered technological evolution there is an increasing need to store patient monitoring data in Databases (DB), allowing the structuring of the data according to the intended application. One of the databases classification is: <u>local DB</u> and <u>remote DB</u>. The local DB is a database created on the device itself and while the remote DB is created on another device (server), to be accessed remotely.

Mobile devices that use a mobile operating system such as Android OS or iOS are characterized by the possibility to store data from mobile applications or even to save records created by applications. Currently, smartphones already have very interesting amounts of internal memory, however there are certain applications that require a large amount of data storage, then to have access from different places and to avoid losing data, thus using remote DBs.

Remote servers allow multiple devices simultaneously to access data in real time, requiring an Internet connection. You must evaluate the system requirements to determine the use of a local or remote database.

Some examples of software technologies for database management are:

- SQLite [49]: It is an Open Source service that allows you to create a database and embed it in various systems. It is used on thousands of systems and uses Structured Query Language (SQL) syntax. It is a service is widely used in mobile applications in for creating local databases.
- MySQL [50]: is a DBMS (Database Management Service) and uses SQL as an interface. It is one of the most popular tools in the world due to its features such as portability, compatibility, low demand for hardware features, ease of use, among others. It is also used on many large-scale sites, including Google, YouTube, Twitter, Facebook, NASA, among others.

Other examples of just plain common usage are PostgreSQL, MongoDB, MariaDB, Oracle, SQL-Server and others.

Most of the time, databases store confidential and personal data of users, such as names, addresses, contacts, e-mails, so this information must be encrypted to comply with security and privacy requirements, either at company level or of users.

The use of databases located on a remote server, like in web services or other external services to save resources on mobile devices, opens new doors for the development of distributed mobile applications, thus creating an opportunity for the medical sciences, namely in the field of physiotherapy and rehabilitation.

The possibility for area professionals to have a mobile application to query information stored in the remote database or even to use another external service dedicated to the area in question, becomes a complementary tool for rehabilitation queries.

2.4 Mobile Applications

A mobile application, commonly known by its abbreviated to APP, is a software designed to be installed on a mobile electronic device such smartphone or tablet. This application can be installed on the device if the device allows you to download it through an online store, such as Google Play, Apple App Store or Windows Store.

Originally the Mobile Applications were created and classified as tools to support the productivity and the recovery of generalized information, including e-mail, calendar, contacts, stock market, meteorological information among others of the genre. However, the growing demand, the ease of availability and the evolution of the Apps, has led to the rapid expansion to other categories, such as Games, GIS, meteorological information services, various order tracking services, ticket purchase, attendance confirmations, connections in social networks, applications in the most diverse areas, such as health, sports, banking and business, stock markets, etc., all this for the majority of mobile devices. The spread in the number and variety of applications served as a stimulus to research and consequent creation of countless Apps to meet the most diverse needs of most utilities.

One of the areas where apps have enormous potential is healthcare. According to [51], mobile devices can help in maintaining a good health status. The APPs developed for this purpose may have features that help improve accessibility to treatments as well as the speed and accuracy of diagnostic exams [51]. Apps can have features that bring patients closer to caregivers or can help with things as simple as adhering to therapy, making the patient forget to take medications.

There are a few good examples of mobile apps that are excellent physical user monitoring tools, feasible to be integrated with remote healthcare services. This can be mentioned:

- ***Google Fit*** [52]: Monitor the effort during physical activity. While walking, jogging or cycling throughout the day, on your phone or watch Android Wear automatically records your activity. It is possible to obtain immediate statistics, in real time of races, walks and bicycle rides. Google Fit records speed, pace, route, elevation, and more, so you can stay motivated and on track (see Figure 2.26).

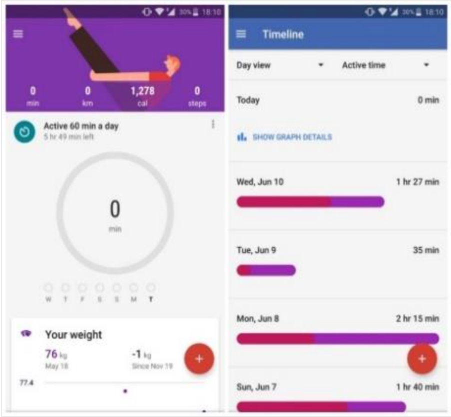

Figure 2.26 - Google Fit App [52].

- ***Nike+ Run Club*** [53]: It monitors your races, establishes challenges, maintains a close network of contacts, evaluates time averages, increases motivation, keeps the user active (see Figure 2.27).

Figure 2.27 - Nike+ Run Club.

The latest developments in wireless sensors networks and cloud technology are providing solutions that can transform our cities into more connected, sustainable and collaborative environments. Recently, wearable devices are emerging and forming a new concept – Wearable IoT (WIoT) because of their ability to detect, transmit and analyze the data. The next generations of WIoT promise to revolutionize the healthcare industry, where individuals are easily monitored by mobile sensors for personalized health and wellness information - vital body parameters, physical activity, behaviour and other critical parameters that affect the quality of everyday life [54]. These technologies are designed to provide better physiotherapist/patient communication, to deliver more efficient care and enable remote patient monitoring as part of independent living solutions. Interconnections found in WIoTs between portable sensors, mobile phones and medical infrastructure allow efficient communication between physicians and patients, allowing them to discuss digitally micro-intervention interventions, comments on symptoms and adaptability to new treatments [54].

Nowadays, doctors want to monitor patients beyond the physical limits of the clinic. In addition, they want to make their patients more proactive regarding their health and medical conditions. Some development platforms, like Kaa Project [55] have emerged to complements wearable technology with superb ready-to-use IoT functions and applications. Kaa is an open source IoT platform built on a modern native cloud architecture and a fully customizable set of features. Based on flexible micro-services, the Kaa easily adapts to almost all needs and applications. It ranges from a small start-up to a large enterprise and supports advanced deployment models for multicloud IoT solutions. Its use cases cover the most diverse areas of industry (see Figure 2.28).

Figure 2.28 - Kaa Project IoT Use Cases.

In order to create a versatile mobile application that could run any mobile device such as smartphone, tablet and laptop, it was necessary to investigate and choose a software framework to produce cross-platform application. There are quite a few possibilities of cross-platform frameworks, including Qt, PhoneGap, Ionic, ReactNative, Python-Kivy or Xamarin.

For a piece of software to be considered cross-platform, it must be able to run on more than one computer architecture or operating system. Developing such a program can be a time-consuming task because different operating systems have different application programming interfaces (API). For example, Linux uses a different API for application software than Windows does.

Web applications are typically described as cross-platform because, ideally, they are accessible from any of various web browsers within different operating systems. Such applications generally employ a client–server system architecture and vary widely in complexity and functionality. This wide variability significantly complicates the goal of cross-platform capability, which is routinely at odds with the goal of advanced functionality.

Basic web applications perform all or most processing from a stateless server, passing the results of the processing to the client web browser where all user interaction with the application consists of simple exchanges of data requests and server responses. These types of applications were the norm in the early phases of World Wide Web application development. Such applications follow a simple transaction model, identical to that of serving static web pages. Today, they are still relatively common, especially where cross-platform compatibility and simplicity are deemed more critical than advanced functionalities.

Web applications are becoming increasingly popular, but many computer users still use traditional application software which does not rely on a client/web-server architecture. The distinction between traditional and web applications is not always clear. Features, installation methods and architectures for web and traditional applications overlap and blur the distinction. Nevertheless, this simplifying distinction is a common and useful generalization.

Chapter 3 - System Description

Overview

This chapter presents all embedded system as well the aspects related the operations associated with the Physio Wear system blocks and the activities associated to the system users.

The Physio Wear system (Fig. 3.1.) is composed of a set of software tools and hardware that support the collection and processing of data, which combines a set of wearable devices for use in natural interactions with a set of highly interactive serious therapeutic games developed in Unity 3D gaming platform.

The system contents a set of smart gloves to be used for natural interactions with the therapeutic serious games for upper limb rehabilitation. In conjunction with the smart gloves, a similar module was developed, which will be applied to a headband, designed to be used as a first-person controller. It will be responsible for collecting the patient's head rotation value, which will be used as the player's avatar head rotation input, approaching the user and the virtual environment in a semi-immersive way. These games provide to the patients with motor impairments the possibility to perform exercises in a highly interactive way based on different VR scenarios. The obtained data can be consulted in a mobile app with goal to help the physiotherapist to evaluate the patients' performance throughout based on metrics visualization associated with performed physical rehabilitation sessions.

The Figure 3.1 illustrates the system architecture that is divided into blocks.

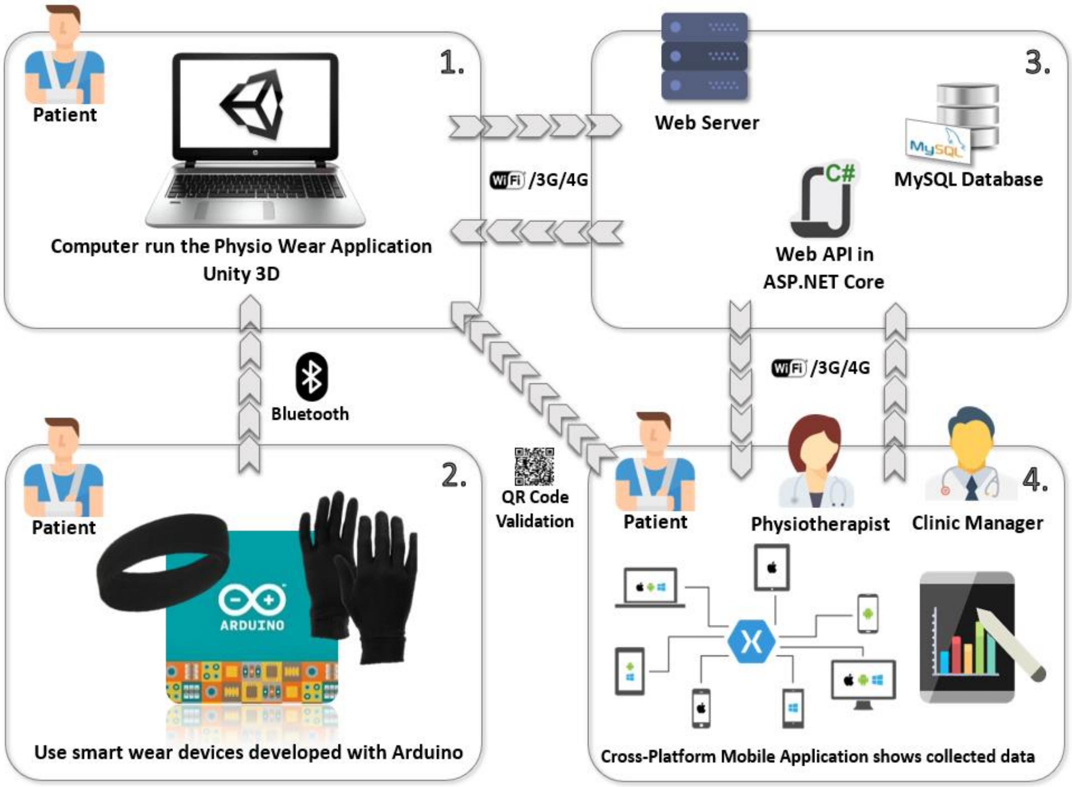

Figure 3.1 - The Physio Wear system architecture.

The block **1.** is dedicated to the patient allowing the access to a set of serious games characterized by natural interaction using Wearable Devices.

The block **2.** is composed by the wearable devices developed with Arduino Platform to interact with games.

The block **3.** it is mainly represented by the data server that stores the game results and game settings. The software technologies used on the server side are MySQL and the Web API in C# that allows the communication between applications.

At the last the block **4.** is expressed by a mobile application associated with patient data management and the health professionals' data management. The implemented IoT architecture block diagram is presented in Figure 3.2.

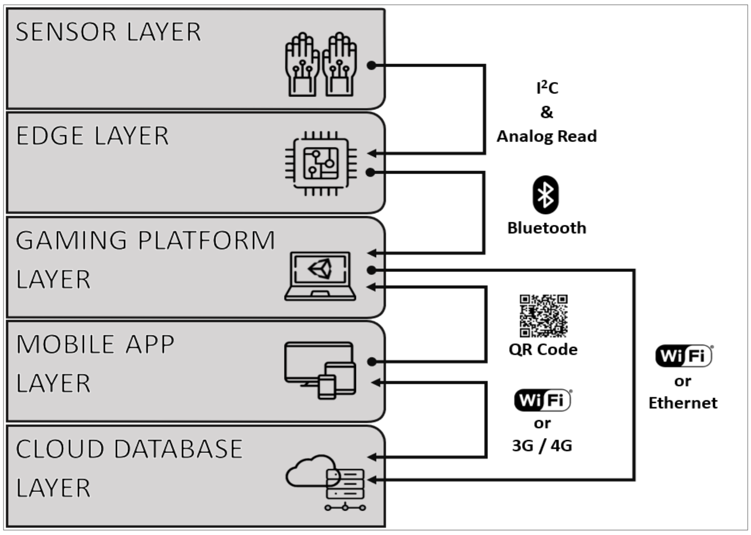

Figure 3.2 - The IoT architecture block diagram.

In Figure 3.2 the following topics are considered:

- The **Sensor Layer** is composed of the IMUs (MPU-9250) and the analog sensors (FlexSensors 2.2" and FlexiForce A201) applied to both gloves.
- The **Edge Layer**, composed of Arduino Nano microcontrollers, after calibration, acquires IMU data, through I²C (Inter-Integrated Circuit) and analog sensors through the AnalogRead method. This data is sent by Bluetooth protocol to Gaming Platform Layer.
- Data generated by the Game, expressed by scores, angles of movement, or feedback force fingers as well as those that arrive at the **Gaming Platform Layer**, via Bluetooth from the sensors are processed and saved in a cloud server in the Cloud Database Layer. Internet connectivity is required in this case.
- The **Cloud Database Layer** will store all data, from the user profile accounts created by the Admin 'system or by the medical staff in the Mobile Application, as well as all exercise monitoring data from the Gaming Platform.

To access the Game profile, for each patient, was developed a **Mobile App Layer**, responsible for performing profile validation on the Gaming Platform, through a unique QR Code generated, for each patient. This QR Code contains all the information necessary to configure the Game and the parameters to be evaluated by the physiotherapist in each training session. This Layer will be responsible for the analysis, processing and visualization of all the data. This process requires a bidirectional Wi-Fi, 3G or 4G connection.

3.1 Users and Applications

The system is described by having three different types of users:

- *Clinic Administrator,*
- *Physiotherapist,*
- *Patient.*

Patient will use wearable devices to interact with the VR scenario to perform the training session imposed by Physiotherapists as part of the patient's rehabilitation plan. The Clinic and Physiotherapist manager interacts with the system through the mobile application. This application is used to manage the available resources, activities such as managing profiles of physiotherapists and patients are considered, definition of training plans, data analysis and visualization, training evaluation report generation. The patient or the accompanying person should use the mobile application to validate the accesses (through QR Code) to the exercises plans as well to access the results and generated reports for each training session of physical rehabilitation.

Two applications were developed as part of the Physio Wear system:

- Computer Application: Developed using Unity 3D gaming platform that serves as a serious gaming platform for patients to complete their training plan. The serious game platform provides a set of highly interactive and configurable games according to the needs of patients. With the aid of developed wearable devices, it is possible to replicate hand movement in the virtual reality scenario as well as head rotation with real-world interaction;
- Mobile Application: was developed using Xamarin cross-platform, which allows to run the same application on computers, tablets and smartphone (with different operating systems iOS, Android OS or Universal Windows Platform) that serves as a work tool for Physiotherapists that allows to add new patients to the system, manage different sessions of physiotherapy and visualize results. The manager also uses this application to add new physiotherapists to the system.

Both applications require an Internet connection to perform the communication with the remote database. Some additional details are mentioned for system users:

- *Patient*: It is the main user of the Physio Wear system, and in this case, it can be mentioned that the developed system corresponds mainly to the patient under physical rehabilitation. The patient represents a single user of the computer application and he is the only one who performs the training created by the user's physiotherapist. The patient can check their results through the mobile application if he received this right form the administrator part.

- *Physiotherapist*: Supervises the training of the patients and creates the training plan. Using the mobile application, physiotherapists enter patient data, create the training plan and analyze the results of the workouts;
- Clinic *Administrator*: Is responsible for managing physiotherapist data using the designed mobile application.

3.2 System Flow Overview

Figure 3.3 illustrates the time flow of the usage of the Physio Wear system by the main users. The roman numeration (I. to XI.) indicates the execution steps. The first step on system usage is carried out by Clinic Manager that introduce Physiotherapists data into the system using the mobile application. The registered Physiotherapist through the mobile application, can introduce the Patients data in the system and to include the training plan according to the patient needs. The Patient will execute the prescribed plan including serious game session using the wearable devices interacting with the computer application.

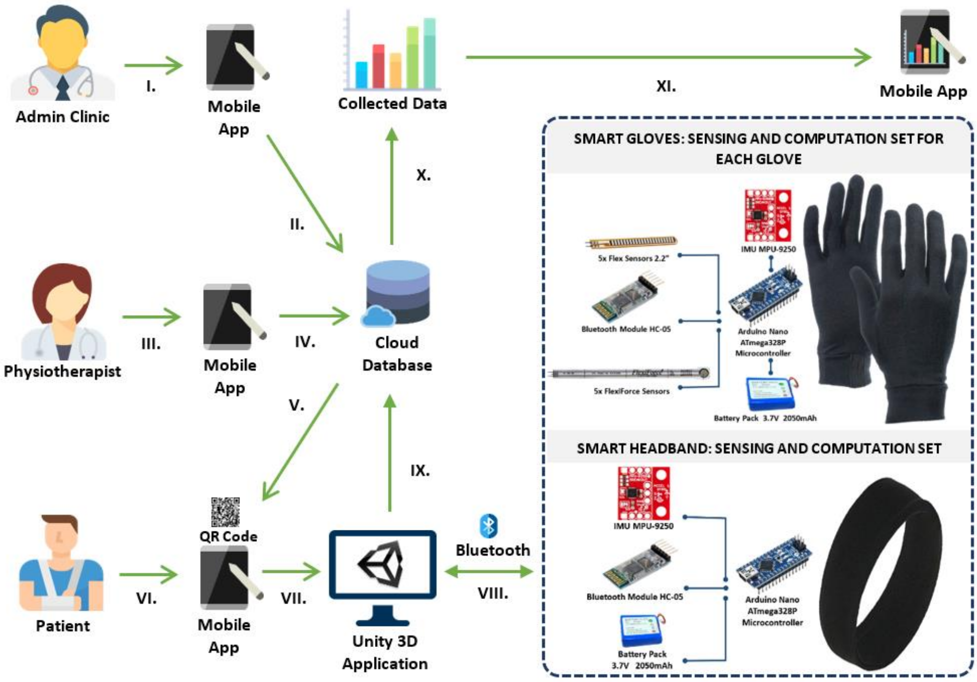

Figure 3.3 - Diagram show time flow of the Physio Wear system.

In Figure 3.3 are indicated the execution steps:

I. Create and Update Doctor Profiles;
II. Settings sent to the Cloud Database;
III. Create and Update Patient Profiles; Selection of the game and configuration of the parameters to evaluate by Physiotherapist; See all results of tasks;
IV. Settings sent to the Cloud Database;
V. Database sends the necessary data for the configuration of the game;
VI. Use Mobile App to access Unity 3D Serious Games via QR Code, with all parameters set; See all results of tasks.
VII. Play the Therapeutic Serious Games with Gloves and Headband;
VIII. Arduino collects sensor data, send to game via Bluetooth Module;
IX. The data collected from the game is stored in the Cloud Database;
X. All data collected are processed to assist a better medical report;
XI. The Physiotherapist gives a report to the Patient via Mobile app.

When a new patient is created by the Physiotherapist in the mobile application, it is recorded that the physiotherapist will be responsible for creating exercise plans for the said patient. However, it will not be an impediment that in the future is implemented a possibility that other physiotherapists may come to create exercise plans for other patients, being responsible only for each exercise plan created there is the possibility of another physiotherapist being able to follow the patient. Concluding the patients may be shared among different physiotherapists in the same clinic.

After adding a new patient into the system, the patient can access the mobile application where they will have access to a unique QRCode to validate access to the exercises in the Unity3D computer application. Figure 3.4 shows an example of the patient profile page of the mobile application, which allows the patient to log in to the computer application, showing the QRCode on webcam available. The use of a QRCode system was considered against other identification technologies, such as NFC or RFID, considering that the costs and capacity of this technology are easily interconnected with the Unity application, allowing a safe and practical access without the need to write in the keyboard.

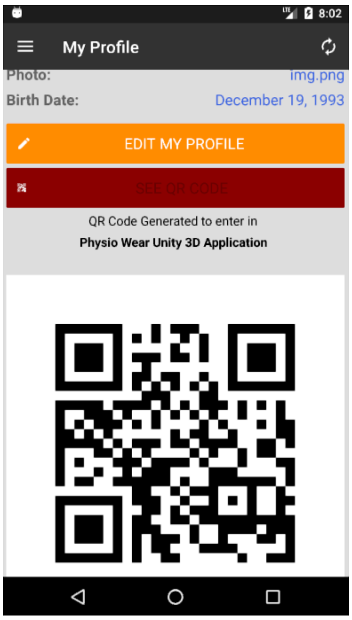

Figure 3.4 - Example of the patient profile page on mobile application, which contains the QRCode that allows the patient to log in to the computer application.

After the patient, is created in the system, the responsibility of the physiotherapist is to analyse patient clinical status and to create an individualized exercise plan. The exercise plan setup involves the following parameters (**III.**):

- *Start Date:* Date for the patient to start the training. The initial date must be equal to or greater than the creation date;
- *End Date:* This date is the last day (inclusive) in which the patient can perform the training. The final date must be equal to or greater than the initial date;
- *Select Game:* What is the serious game that a patient should perform;
- *Duration of the Game:* Game time;
- *Rest time:* Interval time between each game;
- *Number of Repetitions per training:* Number of times the patient should repeat the game;
- *Select Members to Use:* The hand or hands that the patient should use (LEFT, RIGHT, BOTH);
- *Difficulty of the Game:* Level of difficulty of the game (EASY, MEDIUM, HARD);
- *Description:* Training note;

When an exercising plan is created for a patient, the mobile application sends the data to the remote database (**IV.**). Afterwards, the patient should access the computer application presenting the QRCode via mobile app to the webcam, that will read and check the QRCode and check in the DB. If the content belongs to a patient, the patient has successfully logged into the application (**VI.**).

The computer application initiates an HTTP request through the corresponding Web-hosted API on the server so that it contacts the MySQL database and sends the required data to the computer application according to the patient it is using, using an HTTP response (**V.**).

If the patient has exercise plans and these are included between the initial and final exercise planning dates, the patient can perform the serious games configured with the parameters of the training in question (**VII.**).

The application developed in Unity 3D connects via Bluetooth to wearable devices to collect data and replicate hands and their movements in the game scenario (**VIII.**).

Each training session may have a single or more repetitions, this means that if a given workout has, for example, two repetitions, the patient performs once the exercises of serious game with its duration, after completing, has a rest period until the serious game be running again and after completing your training session ends.

For a simplest and direct action by the patient there will only be one active training plan each time for each patient, this implies that every time a physiotherapist creates a new exercise plan, indicate start and end dates that do not match other exercise plans for each patient. Plans can be for a single day or for consecutive days and can be changed and / or deleted by the responsible physiotherapists in the mobile application. In this way the patient upon accessing the game application will always start the exercise plan valid for the given day. In a future use of the system, it is considered the possibility of including several training plans per patient for the same day.

After completion of the exercise plan, the physiotherapist can use the mobile application to change any parameters in the patient's training, if necessary, and visualize the results obtained (**X.**). The results can be of two types, results in terms of scores and results in terms of coordinates. Scores indicate, for example, how many objects the patient picked up during the game, while the coordinates refer to the hand points that the patient used during the exercise. The coordinates refer to values related to rotations, accelerations, forces applied to the hands and fingers.

Finally, if the physiotherapist deems it necessary, he can generate a report with information related to the patient training (**XI.**), including the exercises already performed and their respective scores. These reports can be generated for the personal use of the physiotherapist or can be made available to the patient via mobile application to inform about their performances during the exercises performed.

The execution steps (see Figure 3.3) are repeated several times, either with other exercises or configurations. It is important that there is a follow-up on the part of the physiotherapist, either at an early stage to explain how serious games are performed or at a later stage, to assess the clinical state and the evolution of the patient in performing the movements of the hands and fingers.

The following two sub-chapters provide information about the hardware components of the deployed system.

3.3 Embedded System

In this chapter we will cover the Sensor Layer and the Edge Layer of the IoT architecture block diagram (see Figure 3.2).

3.3.1 Sensor Layer

Force Sensor

The process of creating the electric circuit to be included in the intelligent gloves involved some research regarding the design and testing of the implemented solution. At the same time were studied the possibilities of current sensors that could be included in the current market and which measurements are necessary to be performed for objective evaluation of upper limb rehabilitation.

In the first step of system device the main quantities to be measured were identified. The identified quantities to be measured were:

- Force (compression and extension);
- Acceleration (tri-axial acceleration);
- Orientation (tri-axial orientation)

For the force a piezo resistive sensor that presents resistance variation caused by applied force were considered. Considering that the analog inputs of microcontroller are voltage compatible the conditioning circuits resistance to voltage converters expressed by voltage divider were considered in this particular case.

Using a reference resistor and the resistive sensor (FlexiForce) and applying a reference input voltage an output voltage dependent on applied force on the sensor. Thus, the resistor closest to the input voltage (V_{in}) we'll call R_1, and the resistor closest to ground R_2. The voltage drop across R_2 is called output voltage (V_{out}), that's the divided voltage our circuit exists to make (see Figure 3.5). The voltage divider equation assumes the three values known of the above circuit: the input voltage (V_{in}), and both resistor values (R_1 and R_2). Given those values, we can use the equation (1) to find the output voltage (V_{out}) [56].

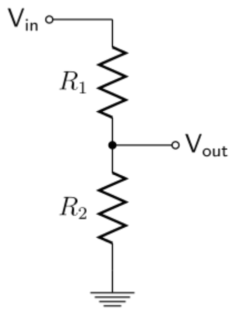

Figure 3.5 - The voltage divider circuit.

$$V_{out} = V_{in} \cdot \left(\frac{R_2}{R_1 + R_2}\right) \qquad (1)$$

Analog to Digital Converter

An **Analog-to-Digital-Converter** (**ADC**) is a very useful feature that converts an analog voltage to a digital number.

The Arduino board includes ADC and a multiplexer module that provides 8 analog inputs labelled (A0 through A7) .to indicate these pins can read analog voltages. ADCs can vary greatly between microcontroller. The Atmel microcontroller of Arduino board includes a 10-bit ADC meaning that it has 1,024 (2^{10}) quantization levels.

The ADC reports a *radiometric value*. This means that the ADC assumes 5V corresponds to 1023 and lower values of the applied voltage will be a digital number between 0 and 1023.

Analog to digital conversions are dependent by the reference voltage. In the Arduino case the reference voltage is V_{ref} 5V that means the digital code "n" for acquired V voltage is given by (2) [57]:

$$n = \frac{V}{V_{ref}} * 1024 \qquad (2)$$

Inertial Measurement System

In order to recreate a virtual environment based on serious games for upper limbs that includes hand and finger movement, an Inertial Measurement Unit (IMU) was included in base circuit. The **SparkFun MPU-9250 IMU Breakout** (see Figure 3.6) was used as the system IMU. MPU-9250 are System in Package (SiP) that combines two chips: the MPU-6500, which contains a 3-axis gyroscope as well as a 3-axis accelerometer, and the AK8963, which features a 3-axis magnetometer. The MPU-9250 uses 16-bit analog-to-digital converters (**ADC's**) for digitizing all 9 axes [58]. By merging the data from these sensors, it possible obtain reliable values for the rotational movements of the real objects by entering those values in the actions on the virtual objects. Figure 3.7 shows the schematic.

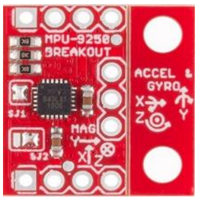

Figure 3.6 - SparkFun MPU-9250 IMU Breakout (true dimensions) [58].

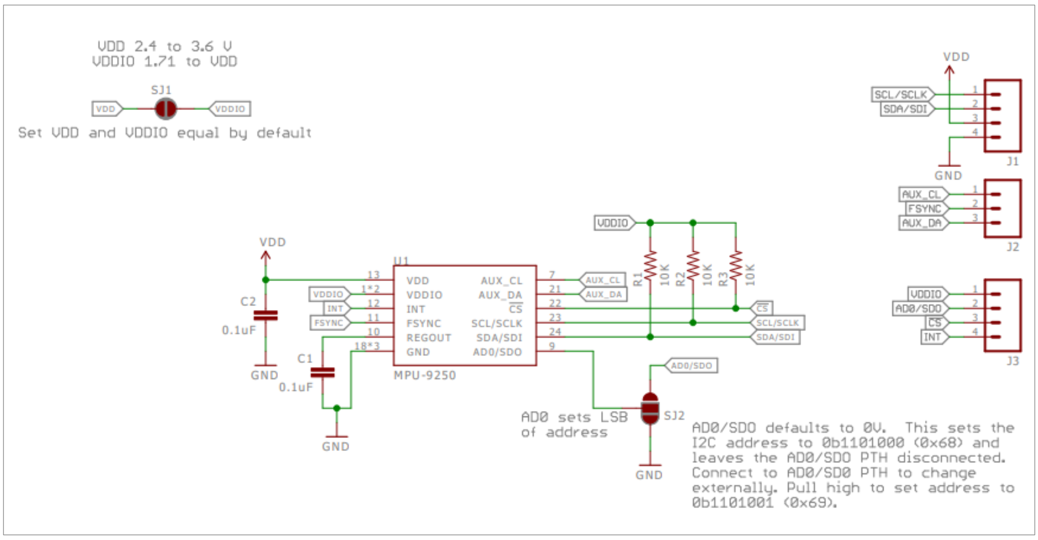

Figure 3.7 - SparkFun MPU-9250 IMU Breakout Schematic [58].

The SparkFun MPU-9250 IMU has following features:

- Digital-output X-, Y-, and Z-axis angular rate sensors (gyroscopes) with a user-programmable full-scale range of ±250, ±500, ±1,000 and ±2,000°/sec and integrated 16-bit ADCs;
- Digital-output triple-axis accelerometer with a programmable full-scale range of ±2g, ±4g, ±8g and ±16g and integrated 16-bit ADCs;
- 3-axis silicon monolithic Hall-effect magnetic sensor with magnetic concentrator;
- Digitally programmable low-pass Gyroscope filter;
- Gyroscope operating current: 3.2mA;
- Accelerometer normal operating current: 450µA;
- Magnetometer normal operating current: 280µA at 8Hz repetition rate;
- VDD supply voltage range of 2.4 – 3.6V;
- Small board design (2 x 2 x 1 inches);
- Detachable mounting holes.

The sensing of the applied force on objects, during the fingers' movements training is based on a set of sensors **Flexi Force A201** (see Figure 3.8) that are included in the system. FlexiForce is a Piezo-resistive force sensor from Tekscan [59]. The characteristics of the sensors is presented in Figure 3.8.

The overall length is about 8.5". Sensor comes with 0.1" spaced, reinforced, breadboard friendly connector. This sensor comes in three models. The sensor ranges from 0 to 1 lb, 25 lbs or 100 lbs (~45 kg / ~ 445 N) of pressure in the mega-ohm range. These sensors are mounted on the glove level, one for each finger.

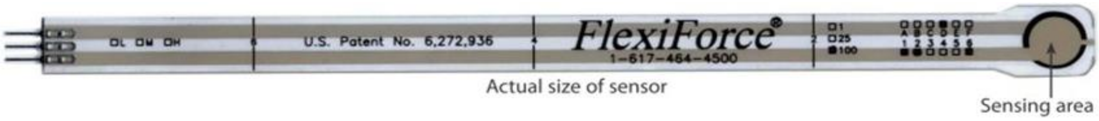

Figure 3.8 - FlexiForce A201 [59].

When no pressure is being applied to the Force Sensitive Resistor (FSR), its resistance will be larger than 1MΩ. The harder you press on the sensor's head, the lower the resistance between the two terminals drops. By combining the FSR with a static resistor to create a voltage divider, you can produce a variable voltage that can be read by a microcontroller's analog-to-digital converter.

The finger flexion and compression of fingers is extracted using **Flex Sensors 2.2"** (see Figure 3.9) patented technology by Spectra Symbol [60]. A simple flex sensor 2.2" in length. As the sensor is flexed, the resistance across the sensor increases. The resistance of the flex sensor changes when the metal pads are on the outside of the bend (text on inside of bend). Connector is 0.1" spaced and bread board friendly. Left flat, these sensors will look like a 30kΩ resistor. As it bends, the resistance between the two terminals will increase to as much as 70kΩ at a 90° angle. By combining the flex sensor with a reference resistor (in this case 33kΩ) to create a voltage divider, you can produce a variable voltage that can be read by a microcontroller's analog-to-digital converter.

Figure 3.9 - Flex Sensor 2.2" by Spectra Symbol [60].

Data acquired from the sensors are primary processed by an Arduino Nano Microcontroller and sent through Bluetooth using a HC-05 Module previously configured as a slave, to a COM serial port of the client platform where the Serious Therapy Game Application developed in Unity 3D runs. The **Bluetooth Module HC-05** [61] (see Figure 3.10) is easily integrated with Arduino boards and enables the option of wireless Bluetooth communication using the serial port. The Bluetooth Module presented also enables serial communication through other pins, so your Arduino has two or more serial ports available. To do this, simply use the *SoftwareSerial.h* Arduino library and configure the module's jumpers. With the Bluetooth Module, any communication between an Arduino and a smartphone or computer can be made. It has a range of up to 10 meters without obstacles and supports Slave and Master use profiles. In Master Mode: The module can connect to other Bluetooth devices. In Slave Mode: The module only receives connections from other Bluetooth devices [61] The Figure 3.11 shows the HC-05 schematic.

Figure 3.10 - Bluetooth Module HC-05 [61].

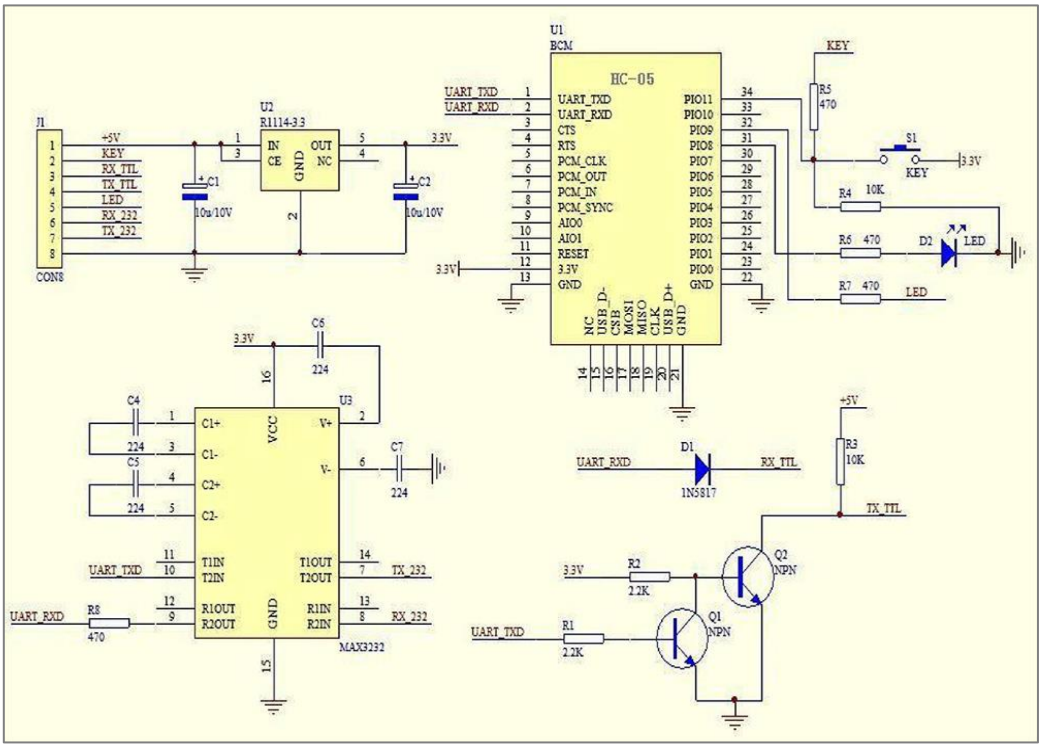

Figure 3.11 - Bluetooth Module HC-05 Schematic [61].

3.3.2 Edge Layer

Arduino an electronic prototyping platform, designed with an Atmel AVR microcontroller with built-in input / output support, a standard programming language, which originates from Wiring, and is essentially C / C ++. The goal of the project is to create tools that are affordable, inexpensive, flexible and easy to use by beginners and professionals.

The Arduino platform can be used for the development of independent interactive objects, or to be connected to a host computer. A typical Arduino board consists of a controller, a few digital and analog I/O lines, and a serial or USB interface to interface with the host, which is used to program and interact with it in real time. The board itself does not have any networking capability, however, it is common to combine one or more Arduinos using appropriate communication boards.

For this project, the data acquisition is performed using **Arduino Nano Microcontroller** due to its higher autonomy (Power Consumption: 19 mA), reduced dimensions (PCB size: 18mm x 45mm) and very light (Weight: 7g). The Arduino Nano is a small, complete, and breadboard-friendly board based on the ATmega328 (Arduino Nano 3.x) (see Figure 3.12). It works with a Mini-B USB cable instead of a standard one [62] and presents the following characteristics:

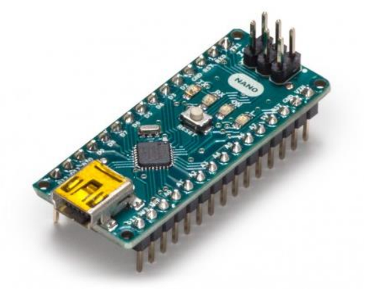

Figure 3.12 - Arduino Nano Microcontroller board [62].

- Operating Voltage: 5V
- Flash Memory: 32 KB of which 2KB used by bootloader;
- SRAM: 2 KB
- Analog IN Pins: 8
- EEPROM: 1 KB
- DC Current per I/O Pins: 40mA (I/O Pins)
- Input Voltage: 7-12 V
- Digital I/O Pins: 22 (6 of which are PWM)
- PWM Output: 6

Figure 3.13 shows the set of all sensors that are used for each glove. Figure 3.14 presents the implemented circuit diagram for each smart glove, with all their components.

Figure 3.13 - Smart gloves: sensing and computation set.

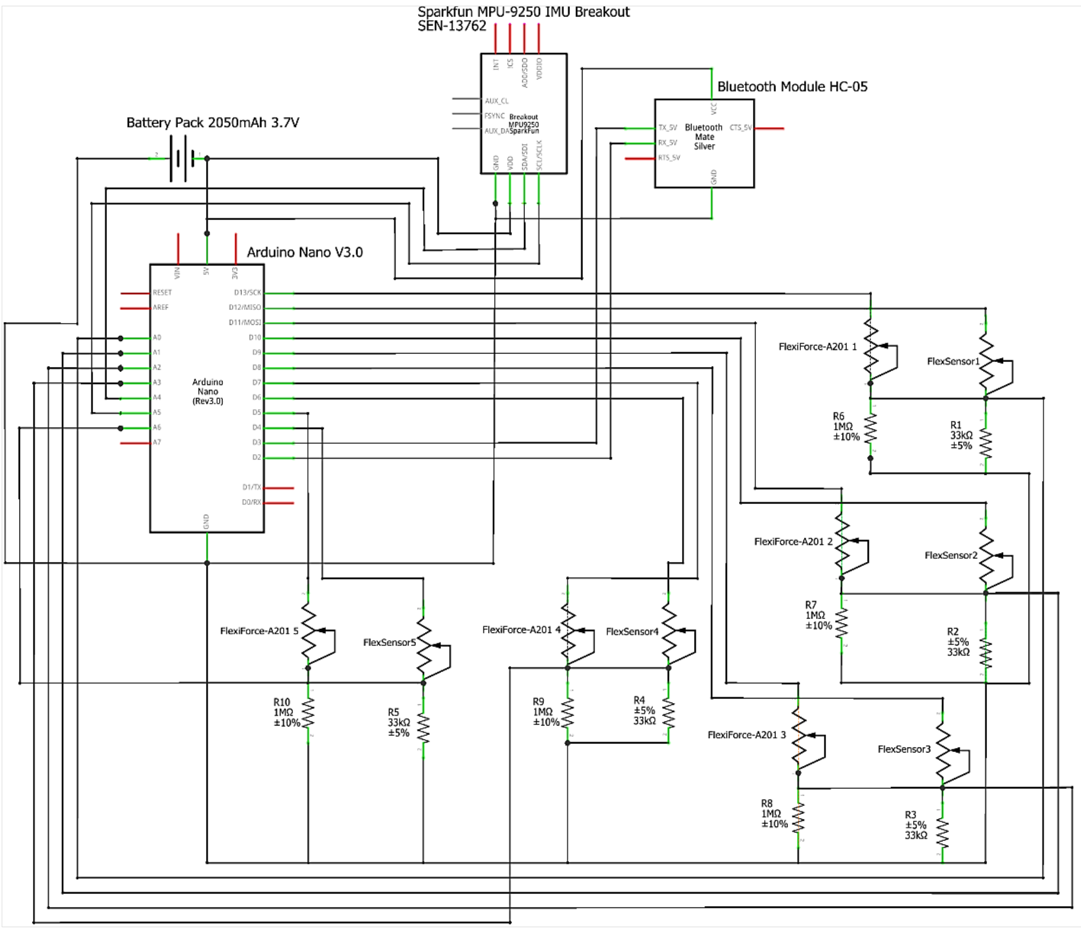

Figure 3.14 - The implemented circuit diagram for each smart glove.

A Printed Circuit Board (PCB) was designed to make the circuit as small as possible. Figure 3.15 presents the implemented smart glove prototype, including the force sensors placed on the fingers, on top of the glove (only for perception) and the signal conditioning, primary processing and communication board. Figure 3.16 shows the final mounted prototype of a smart glove for Right Hand.

Figure 3.15 - Hardware components attached on a glove. On outside (only for perception).

Figure 3.16 - Hardware final mounted prototype of a first version smart glove.

A first version of the smart gloves was developed with cloth appliques on a pair of gloves purchased in the current retail market. The final product is shown in Figure 3.16.

After some intensive tests of validation of the analog readings of the piezo-resistive Flex Sensors, and due to the poor measurements obtained, it was decided to develop a pair of gloves adapted to the sensor sets to be applied. In this way a pair of gloves with spaces (tissue channels) on each of the fingers were designed and produced in lycra fabric to allocate the FlexiForce A201 sensors on the underside of each finger, and even for the FlexSensors 2.2" on the upper face. The Figure 3.17 presents the second version already with the sensors applied.

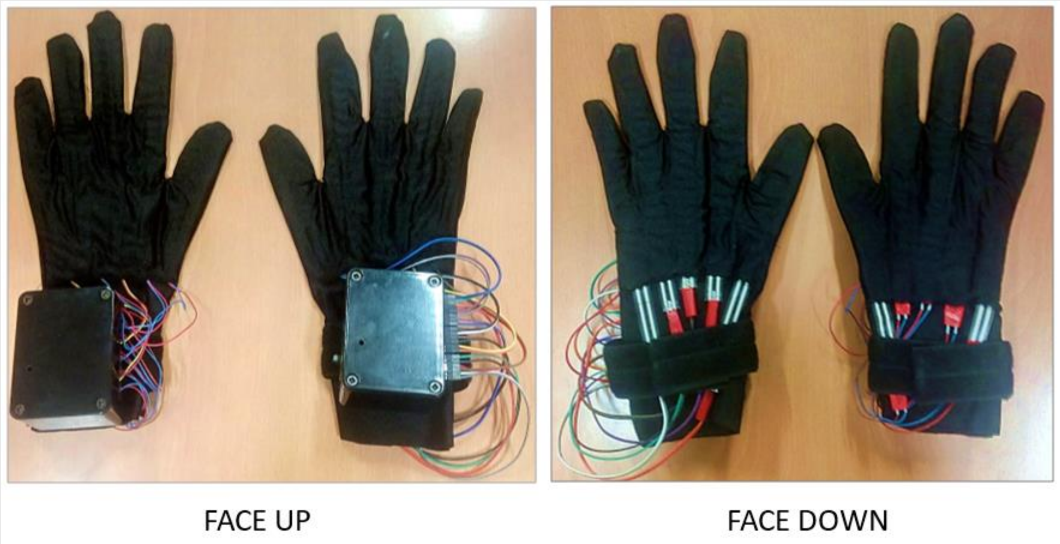

Figure 3.17 - Hardware final mounted prototype of a second version smart gloves.

With the goal of making physiotherapy exercises easy to follow in gaming mode and highly interactive with the user, several Virtual Reality scenarios for the Gaming Platform are designed. Thus, the implemented system permits to animate an Avatar using a First-Person-Controller (user as controller).

This controller method allows the game scene Camera to be placed in the position of the Avatar's eyes. On Classic PC Games, that use First-Person-Controller this player construction process is usually used with the mouse cursor, where the rotation inputs of the Avatar are generated by the cursor movements. In the present case considering the necessity to interact with different scenarios to improve the upper limb capability, the mouse cursor readings were replaced by the rotation movement readings captured and validated by the IMUs. A headband will be responsible for collecting the head rotation value of the patient, which will be used as the rotation input of the player's avatar head. The same application is performed with smart gloves data to animate the members.

Figure 3.18 shows the set of sensors that are used for headband.

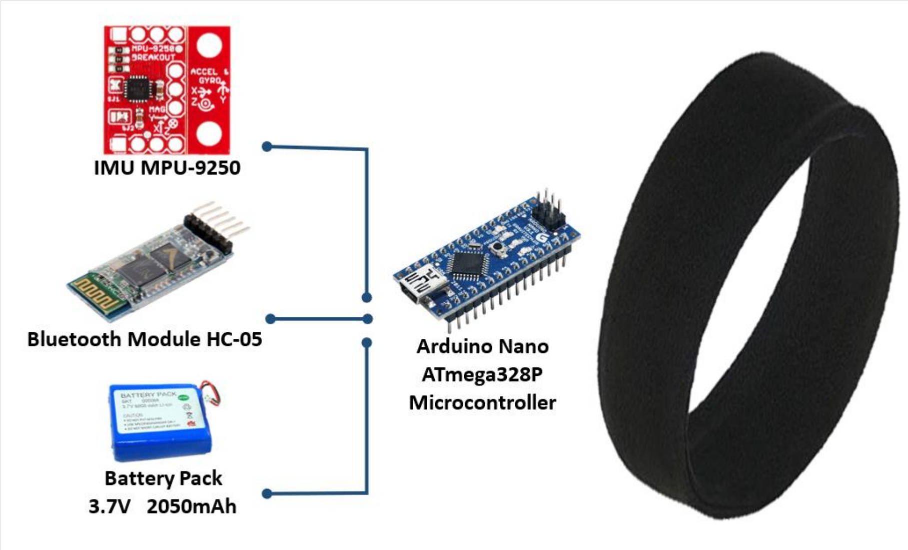

Figure 3.18 - Smart headband: sensing and computation set.

Figure 3.19 presents the implemented circuit diagram for headband with all their components.

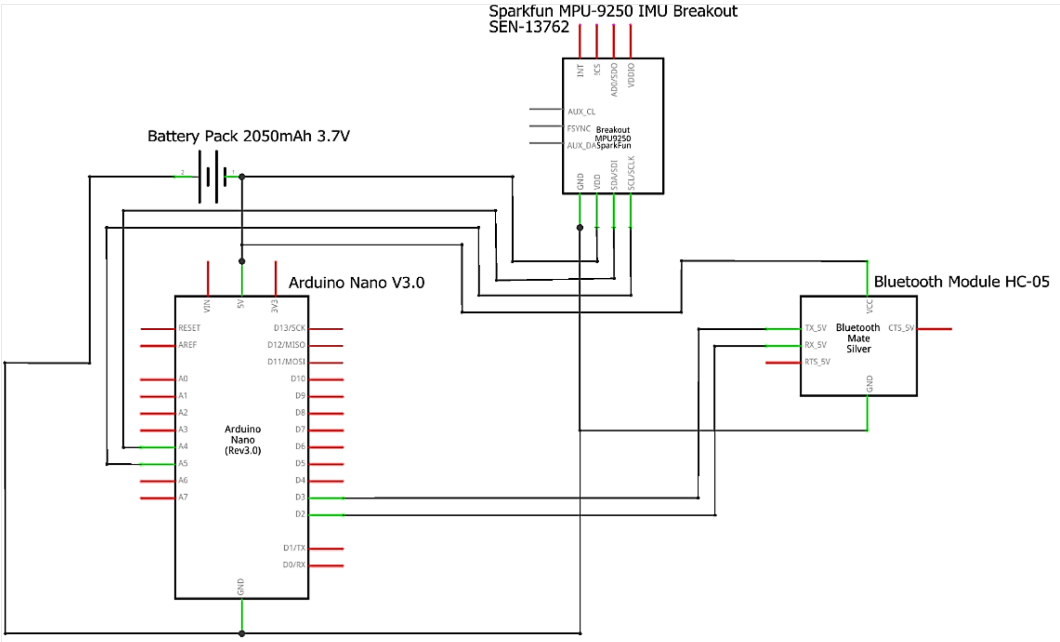

Figure 3.19 - The implemented circuit diagram for smart headband.

Figure 3.20 shows the final mounted smart headband set.

Figure 3.20 - The final mounted set to smart headband.

The electric circuit boxes are attached to the wrists and to the headband through a Velcro tape, so that it is possible to adjust them in the best way to the members.

The IMU modules (particularly magnetometers) for their use, imply that they are calibrated at the place where the exercises will occur before any type of data acquisition is performed. This is necessary because of the variations of magnetic field verified from place to place. This procedure only needs to be done once per location. It is a little demanding on a technical level since it involves placing in the Arduino devices a script (from a library) to acquire magnetic field readings while the devices are rotated in all directions. When there were no variations in the readings, the data is written to the internal memory of the devices, in order to construct reliable values of the rotational movements using the merge algorithms of the raw data. This process is described in more detail in the User Manual in Appendix B.

3.3.3 Embedded Software

The embedded software associated with signal acquisition coming from the sensors and the primary processing of the digital data was carried out development C scripts under the Arduino IDE platform.

The embedded sensor was carried out using several libraries associated with analog readings Bluetooth module configuration libraries, and data acquisition and processing libraries of the IMU module.

The usage of MEMS IMU [63] including accelerometers, gyroscopes and magnetometers can be used to obtain accurate orientation measurements through sensor fusion algorithms. MEMS IMUs are vulnerable to various kinds of noise and errors. Accelerometers are extremely sensitive to attitude changing and impact forces while gyroscopes are sensitive to temperature changes and suffer from a slow-changing bias. To summarize, accelerometers have poor dynamic features and gyroscopes have poor static features. Therefore, an Attitude and Heading Reference System (AHRS) algorithm is needed to fuse the data from different sensors to overcome the drawbacks of each of them and take the most reliable part from them respectively to give a best prediction of the sensor actual status.

AHRS algorithm is the foundation of position estimation [64] because gravity must be removed completely from the accelerometer to get linear acceleration. Only then can the integration be done without being concerned about the drift. Otherwise small errors in the measurement of acceleration are integrated into progressively larger errors in velocity, which are compounded into still greater errors in position. Since the new position is calculated from the previous calculated position and the measured acceleration and angular velocity, these errors accumulate roughly proportionally to the time since the initial position was input.

Like the accelerometer, gyroscope also has the integration drift problem-small errors in the measurements are dramatically amplified by integration over time. However, gyroscopes suffer from another kind of drift. Gyroscopes are sensitive to changes in temperature, which brings about slow-changing deviations just like integration drift did for the accelerometer [65].

The first thing obtained is the roll and pitch information. The convention used, if the device is an aircraft for example, then the x-axis points forward from the nose as seen by the pilot, the y-axis points out along the right-wing and the z-axis points straight down. So, a positive roll angle means that the right-wing rolls down, a positive pitch angle means that the nose points up and a positive yaw angle means that the nose rotates towards the right (see Figure 3.21). The name of these measures is Euler Angles [66].

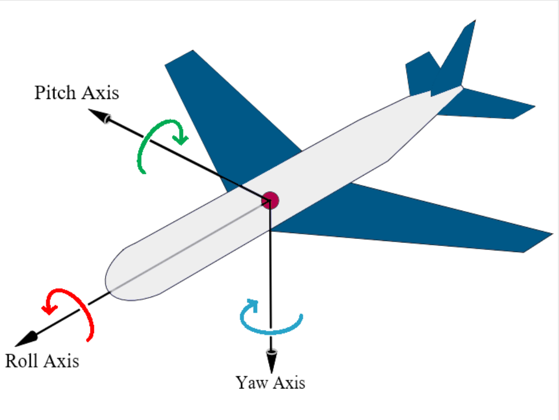

Figure 3.21 - Euler Angles convention [67].

In the case, AHRS algorithm implementation, we base the sketches in the available libraries for a set of many IMUs at richardstechnotes' github, RTIMULib-Arduino [68] and kriswiner' github library [69]. These libraries support the correction of arbitrary, non-standard, coordinate axis-aligned IMU orientations. However, it is very often true that an installation of an IMU in any device characterized by a few degrees offsets on each axis.

According the equations 25 and 26 in [70], by utilizing the accelerometer output, the rotations around the X-axis (Roll) and around the Y-axis (Pitch) can be calculated. Thus, if $\mathbf{Accel}_X$, $\mathbf{Accel}_Y$, and $\mathbf{Accel}_Z$ are accelerometer measurements in the X-, Y- and Z-axes respectively, the Roll and Pitch angles are (in radians):

$$Roll = \arctan\left(\frac{\mathrm{Accel}_Y}{Accel_Z}\right) \qquad (3)$$

$$Pitch = \arctan\left(\frac{-\mathrm{Accel}_X}{\sqrt{((\mathrm{Accel}_Y)^2 + (\mathrm{Accel}_Z)^2\ )}}\right) \qquad (4)$$

In order to measure rotation around the Z-axis (yaw), the other sensors need to be incorporated with the accelerometer. The following flow chart shows how it is do it using quaternions (see Figure 3.22).

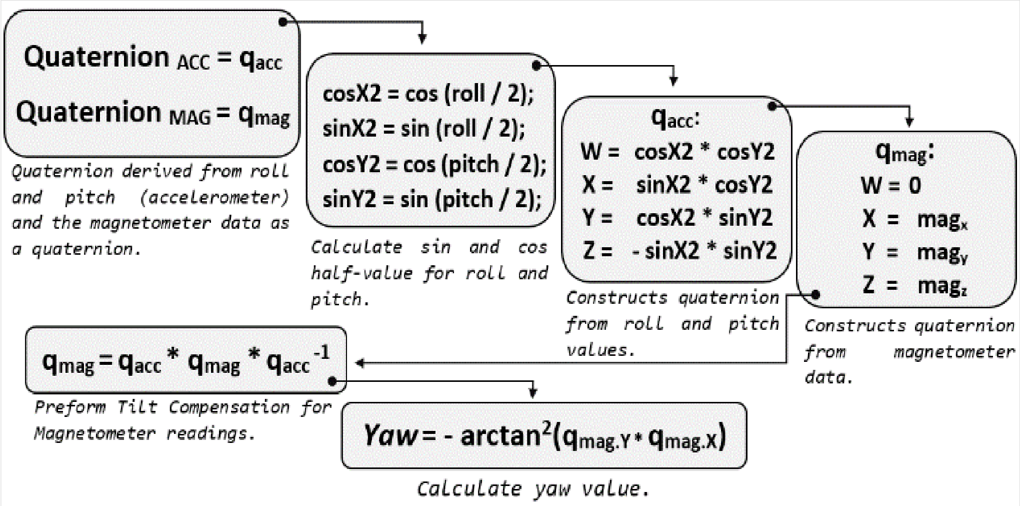

Figure 3.22 - Flow chart showing how calculate Yaw using quaternions.

In Figure 3.23 the first part creates a quaternion *q* representing the roll and the pitch - all that stuff with *sin* and *cos* is how you create a quaternion from Euler angles (yaw is always 0 for this purpose). Next, the magnetometer vector is placed in a quaternion, ready to be tilt compensated. In the line before the last, the magnetometer information is rotated by the quaternion from the roll and the pitch and becomes the tilt-compensated version (that is, the values you would get if the magnetometer were perfectly horizontal). Lastly standard way of generating yaw value is done from tilt-compensated magnetometer readings.

The corrections are done using quaternions, to obtaining reliable values of the Euler angles (roll, pitch and yaw). A Kalman Filter, as can be demonstrated in [71], is used to perform the tilt compensation of the readings obtained from the accelerometer and the magnetometer. For such uses the quaternions to merge with the Gyroscope readings. From this fusion results the AHRS algorithm for reliable object orientation (see Figure 3.23). This data fusion can be found in the RTIMULib-Arduino library [68] in the RTFusionRTQF.cpp file. Tilt compensation is required because the magnetometer reads the magnetic field in the orientation of the object where it is mounted, while the required measurements are in the horizontal plane.

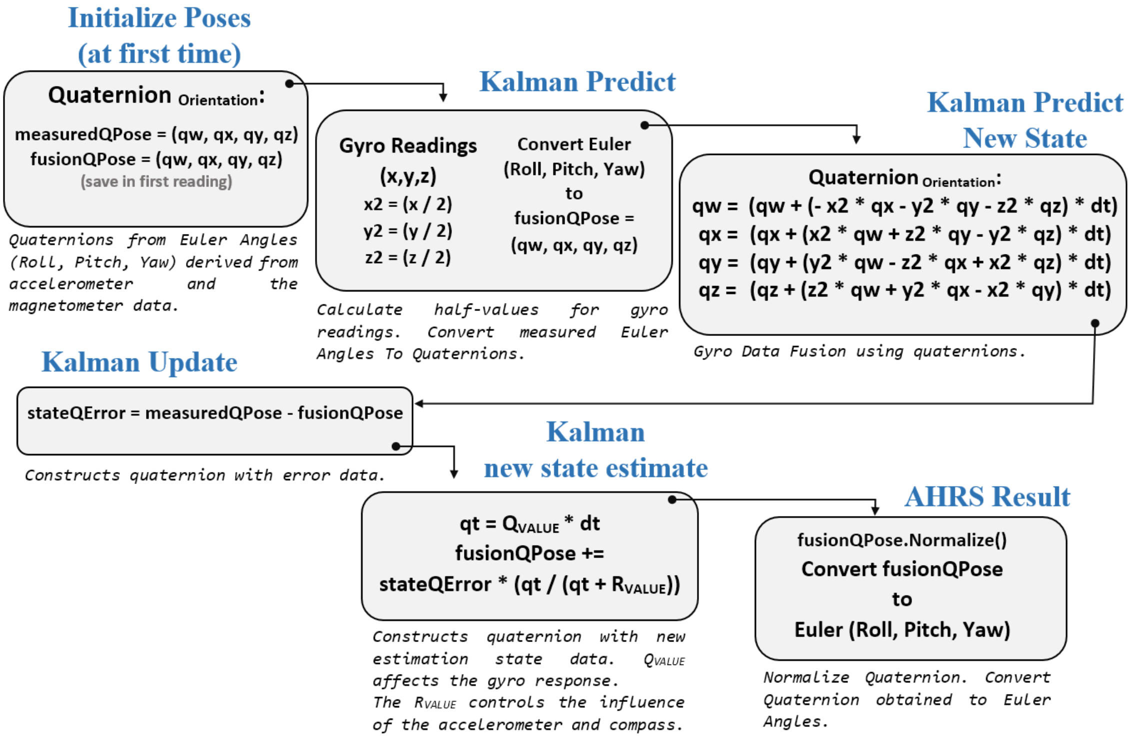

Figure 3.23 - Flow chart showing how create AHRS algorithm using quaternions based RTIMULib [68].

Using the same principle, it is then possible to calculate the Linear Acceleration (or residual, without gravity component). Table 1 shows the math equations in Derivate and Integral Forms that correlate the Acceleration with Velocity and Position.

Table 1 - Math Equations in Derivative and Integral Forms that correlate the Acceleration with Velocity and Position.

	Derivative Form	Integral Form
Acceleration	$a(t) = \frac{dv}{dt} = \frac{d^2r}{dt^2}$	$a(t)$
Velocity	$v(t) = \frac{dr}{dt}$	$v(t) = v_0 + \int_0^t a dt'$
Position	$r(t)$	$r(t) = r_0 + \int_0^t v dt'$

The double integration of residual accelerations with respect to time to give a change in position, its simple in principle, but is not like that. Accelerations can be mathematically integrated once to obtain velocity and twice to obtain changes in position. Only the accelerations due to effective hand motion must be integrated. Since the accelerometer provides a combination of accelerations due to motion, gravity and error, it becomes necessary to separate these acceleration components. Errors in measured accelerations can be deterministic (bias, scale factor and axis misalignment) and/or random errors. Such errors propagate through the integration process, causing a considerable drift in estimated positions. Owing to drift error, position estimation cannot be performed with adequate accuracy for periods longer than few seconds. The **code snippet 1** taken from library [68] shows how obtain Euler Angles and to calculate Linear Acceleration using quaternions and is presented on *attachments*.

In order to be able to read 10 analog output sensors using the Arduino Nano platform, which only has 8 analog inputs, and 2 of them (SDA & SCL) are used by I2C technology for connection to the IMU module, a **Time Multiplexing** is performed on each analog input so that reliable readings of the two analog sensors associated with each finger can be obtained using a single analog input channel. In other words, for each clock cycle, two sensors are read on the same analogue input port.

The Arduino function *digitalWrite* (HIGH) command sets to 1 the digital input value that feeds the electrical circuit of the first analog sensor, then the *digitalWrite* (LOW) command sets this value to 0, a very short time delay (order of milliseconds) is applied and repeat the process for the second analog sensor. This procedure is performed simultaneously in each of the analog ports that will make the data acquisition of piezo-resistive force sensors. The **code snippet 2** show how it is done and is presented on *attachments*.

For each <u>smart glove</u>, the collected measured data are described below:

- The *rotation* (Euler Angles) values and *linear accelerations* of the upper limbs (forearm) are obtained and validated by the IMU module;
- The FlexSensors 2.2" capture the *angular flexion and compression* values exerted by each finger on each hand;
- The FlexiForce Sensors A201 capture the *pressure* values exerted by each fingertip on each hand.

The <u>smart headband</u> including IMU collects the measured data associated with:

- The *rotation* (Euler Angles) values and *linear accelerations* of the head are obtained and validated by the IMU module.

3.4 Another Hardware Components

In addition to the developed wearable devices, there are other hardware devices that are required to use the Physio Wear applications:

- Computer with Webcam and Bluetooth to play serious games;
- Server to store all data;
- Smartphone, tablet or computer with Android OS, iOS or UWP to access the mobile app.

Computer with Webcam and Bluetooth:

The Unity application is designed for computers running the Windows OS or Mac OS. For its development and testing a portable computer with the following specifications was used:

Table 2 - Specifications of the laptop computer used to develop and test the Physio Wear System.

HP ENVY 15-K200NP	
Windows Edition	Windows 10 Home
Screen inches	15,6 "
Processor	Intel® Core™ i7-5500U Dual Core @ 2.40GHz
RAM Memory	16,00 GB
Type of system	64-bit operating system, x64-based processor
Video card	NVIDIA® GeForce™ GTX 850M
Webcam	Webcam HP TrueVision HD
Communications	Wireless LAN 802.11b/g/n/ac Bluetooth 4.0 Rede Gigabit 10/100/1000

The clinics where the system is installed should use computers that have similar or higher specifications, such as those described above, be they desktop or laptop computers. It is recommended that computers have a good video card, with minimum resolution of 1280 × 720, processor and graphics card equal to or greater than those indicated in Table 1, so that the patient can view all the components of the game. Preferably and in order to improve the user experience, the computer application should be performed with the aid of a large-sized monitor (for example, 40" or greater).

About the Webcam (HP Webcam TrueVision HD) is an essential component for the application because it is through it that the patient connects to the application, showing his QRCode that will be accessible in his profile of the mobile application.

The solution presented using QR Code on login will require less effort than compared to the process of writing validation credentials using the keyboard, making the system more practical in its configuration and use by the patient.

Regarding the Bluetooth protocol, it is also important that computers have the ability to communicate via Bluetooth protocol. This requirement is also essential to run the Unity application on the computer because it is through the Bluetooth protocol that will establish the communications between the wearable devices and the game system. There are computers on the market that already have this built-in source protocol. In case they do not have it, it is possible to buy this system in the market for low values. At the physical level Bluetooth is characterized by a small Pen-USB, providing this capability of communication to the host computer.

Server:

In this implementation we used a Virtual Private Server (VPS) consisting of a virtual machine. This server has its dedicated operating system and user access rights, which allow it to install any software compatible with the operating system. On the server is hosted MySQL database v5.7.11, making use of the Web API developed in ASP.NET Core v2.1 in C#. Also stored here are the coordinate files resulting from the patient training, profile pictures of the various users and QR Codes for system access.

Mobile Device (Smartphone / Tablet / Laptop):

To use the mobile application requires a device with the Android operating system, iOS or Universal Windows Platform. In the developed Physio Wear mobile application it is possible to visualize graphs referring to the results of the executions performed by the patients, so it is advisable to use tablets, allowing a better visualization of the information made available to the user.

The devices were used for the development and testing of the mobile application, were a Tablet Android Samsung Galaxy Tab S2 9.7” and a smartphone BQ Aquaris A4.5 Android One.

Figure 3.24 illustrates one of two devices used, which has a 9.7-inch screen with resolution of 2048 x 1536 pixels, a total of 263.9 pixels per inch (ppi).

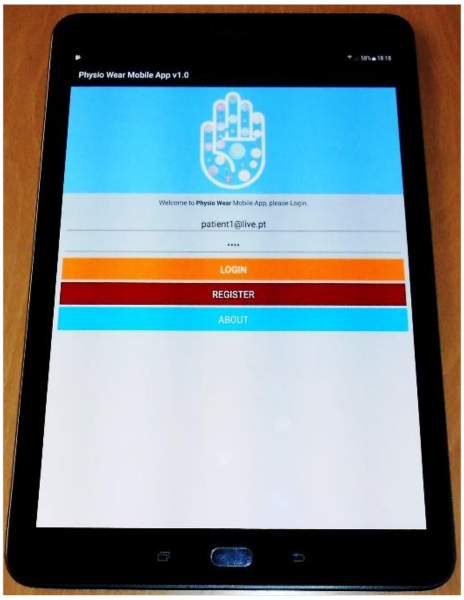

Figure 3.24 - Android Tablet Samsung Galaxy Tab S2 9.7”.

Future devices used by patients, physiotherapists and managers should be tablets with at least the minimum or higher version of the Android operating system, or iPads with the most up-to-date iOS system. The application requires the device to have access to the Internet, either through a mobile network (3G, 4G) or Wi-Fi.

The Table 3 shows the Specifications recommended to the mobile device to run the Physio Wear Mobile App.

Table 3 - Specifications recommended for mobile device to run the Physio Wear Mobile App.

Specifications Recommended	
Operating System	Windows Phone, iOS or Android OS
Screen inches	600 x 1024 pixels or Higher
Processor	Dual-Core or Higher
RAM Memory	512,00 MB or Higher
Storage	1GB or Higher
Communications	Wi-Fi 3G/4G

Chapter 4 - Serious Games for Physiotherapy

Overview

The term Serious Game refers to serious digital games that are based not only on entertainment and fun, but also on games that promote education, health, research, and more. This chapter introduces the computer application that was developed for use by patients so that they can perform the training plans previously created by the physiotherapists. This chapter explores how games were created and adapted to the individual characteristics of each patient as well as the configuration required for communication between clothing devices and the platform.

4.1 Game Engine

The gaming mechanism is a fundamental component in the development of a video game. It is a computer program or a set of libraries, to simplify the development of video games or applications with real-time graphics. There are several gaming mechanisms that can be used to develop 2D or 3D video games, each with certain features to assist the developer in developing, for example, NVIDIA PhysX [23], Unreal Engine [24] or Unity 3D [25]. After a critical review, it was decided to use Unity 3D as a gaming engine because of its excellent rendering capabilities and the features offered. The programming languages used are C # and JavaScript, which makes programming easier because its syntax is identical to other programming languages common among programmers. It is a very well documented game engine with a powerful Asset Store [72] where you can download scripts, objects, textures, among others to develop games, many for free. Unity 3D offers a simple and intuitive interface to help the user to develop the games. It can be personalized and organized according to preferences of developer (see Figure 4.1).

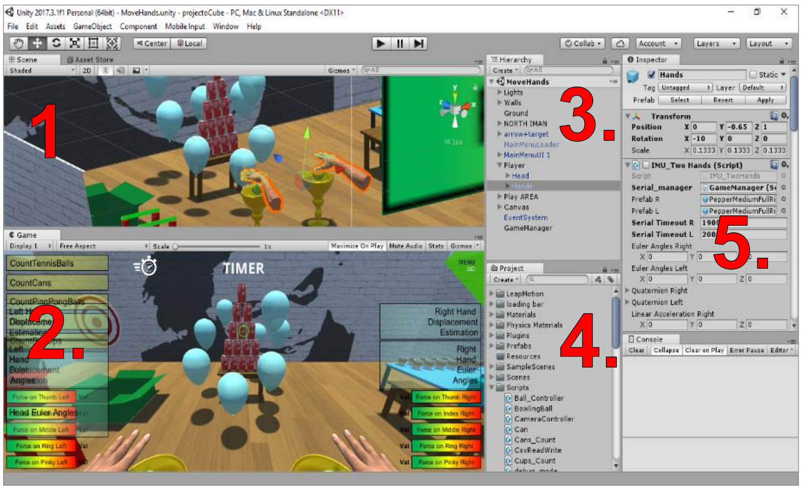

Figure 4.1 - Unity 3D game engine interface.

- **1.** The Scene tab allows the programmer to interact with the objects that are present in the scenario. Allows perform rotations, translations, change its size and apply new textures;
- **2.** Game Engine allows the programmer runs the game to see what has already developed. This is the area where you can see the scenario created in the perspective of the final user.
- **3.** Hierarchy is the area where refers to existing objects in the scenario, where it is possible to remove or add new ones;
- **4.** Project Structure with all the files belonging to the project;
- **5.** Finally, we have the Inspector. For each object is possible to perform several operations, for example, associate scripts or create physics processes. Is very useful to visualize the various components associated with each object. It also has a tab named by Console that is used for debugging processes;

One of the main aspects is the interaction between the scenario and the present objects. Unity 3D doesn’t have any main script to control all the actions of the game, but several scripts that are behaviours and that can be associated with the various objects. After programming a script, it must extend the *Monobehaviour* class and it is ready to be associated with an object, it is only necessary to drag the script to the object, and this is associated as a component. Through the Inspector tool it is possible to view the script as a component of the object and verify if it has, for example, some public variables, is possible to visualize them and to define values for each of them (see Figure 4.2.)

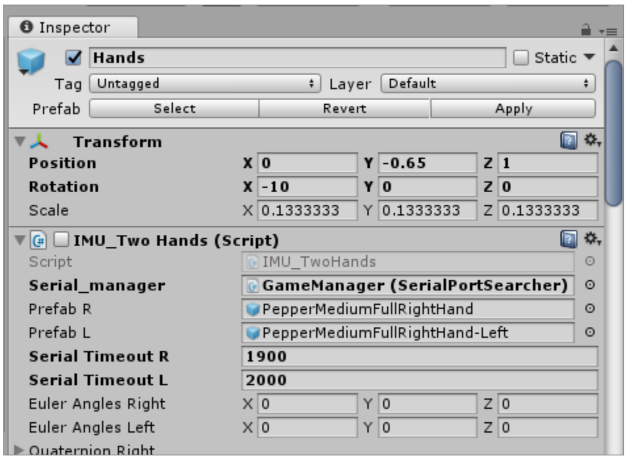

Figure 4.2 - Script inspector example with an object with its attributes and values.

The interaction between objects can be performed in two different ways. The first consists of the laws of physics, for example, when two objects of equal size and different masses collide, the one with the largest mass is thrown from a certain point to the position where the other object is, when it collides with the object it was of static the force exerted on it reacts and its position is changed, and there is a direct interaction between the two objects. Another example can be when an object reacts to a certain light by changing its texture colour.

The second way is that when a given object performs an operation, it sends a notification to another object for it to perform a certain action. There are different types of interactions that help make the scenario as close to reality as possible. The version used to develop the application was Unity 2017.3.1f1 (64-bit).

4.2 First Tests

Initially it was necessary to understand how the communication between the Unity 3D game engine and the Arduino works. Some online tutorials [73] [74] are a good start point and show how it can interact with gaming platform, teaching step-by-step how to create a communication with game engine.

The communication between Arduino and the PC is mediated using the serial port.

The class *SerialPort* is the one that mediates such communication in *C#*. However, Unity doesn’t usually include the necessary libraries to use it. To compensate for this, we need to force Unity to include the full .NET 2.0 library in its executables.

The first tests of communication between the Arduino and the PC were realized with USB cable, and later it was configured the Bluetooth connection, that also is based on a serial communication. Figure 4.3 shows the result of the first game scenario developed where it is possible to visualize an object (red parallelepiped) that receives the calculated rotation values of the IMU from the Arduino and places those values in the entries for rotation of the object.

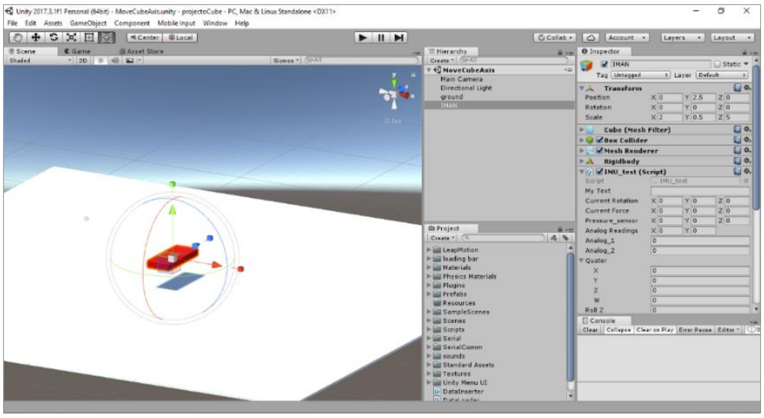

Figure 4.3 - First tests performed on Unity with Arduino Microcontroller and IMU readings.

These tutorials were useful for understanding the interaction between the Arduino and the game engine. The next step was including some object textures to represent Hands (see Figure 4.4). For this purpose, a software suite provided by the Leap Motion Company was used, which provides a Unity 3D project that has several examples [75] to help the programmer create new applications. The project does not just provide examples, it also has different objects representing hands and arms of different colours, genres and sizes, and provides several scripts already programmed and ready for use.

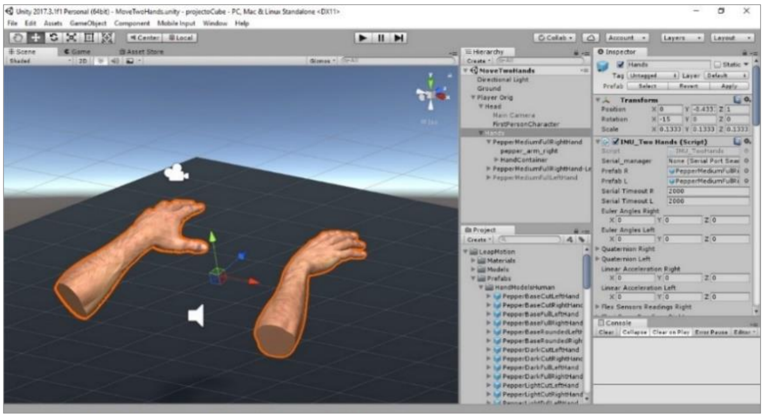

Figure 4.4 - Inclusion of textures related to the objects representing the hands and fingers.

Some methods have been developed in order to allow the representation movements of the virtual objects defined by the fingers. Therefore, each reading obtained from the Flex Sensors 2.2” will act on 3 finger segment joints, allowing the creation of fluid movements to open and close each finger in the virtual scene. These values, then multiplied by a constant scale factor, are placed with a rotation input value at the junction between two segments of each finger (which has 4 segments per finger) (see Figure 4.5). The combination of these methods to act simultaneously in each of the fingers, allows to create interactive virtual movements of the hands and fingers.

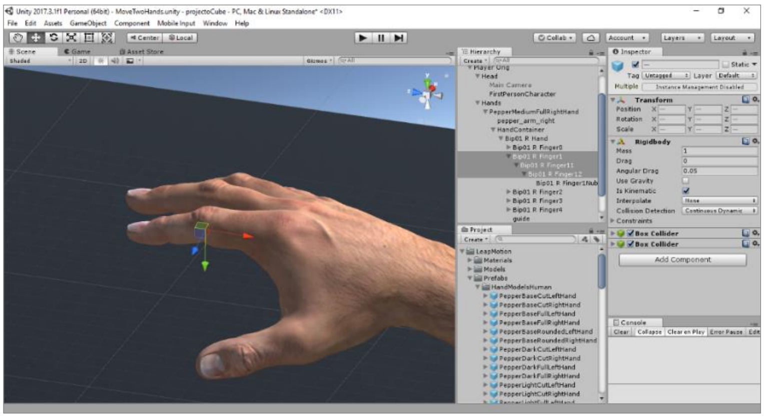

Figure 4.5 - Point of application of the bending values for the index finger of the right hand in the game scenario.

The possibility of interacting with virtual scenario, such as grabbing and dropping virtual objects, implied the inclusion of some *Collider* components in the hands.

Collider components define the shape of an object for the purposes of physical collisions. A collider, which is invisible, need not be the exact same shape as the object’s mesh and in fact, a rough approximation is often more efficient and indistinguishable in gameplay. The simplest (and least processor-intensive) colliders are the so-called primitive collider types. In 3D, these are the *Box Collider*, *Sphere Collider* and *Capsule Collider*. Any number of these can be added to a single object to create compound colliders.

With careful positioning and sizing, compound colliders can often approximate the shape of an object quite well while keeping a low processor overhead. Further flexibility can be gained by having additional colliders on child objects (e.g., boxes can be rotated relative to the local axes of the parent object). When creating a compound collider like this, there should only be one *Rigidbody* component, placed on the root object in the hierarchy [76]. The Figure 4.6 shows all collider components (Green Shape Lines) included in both hands texture objects.

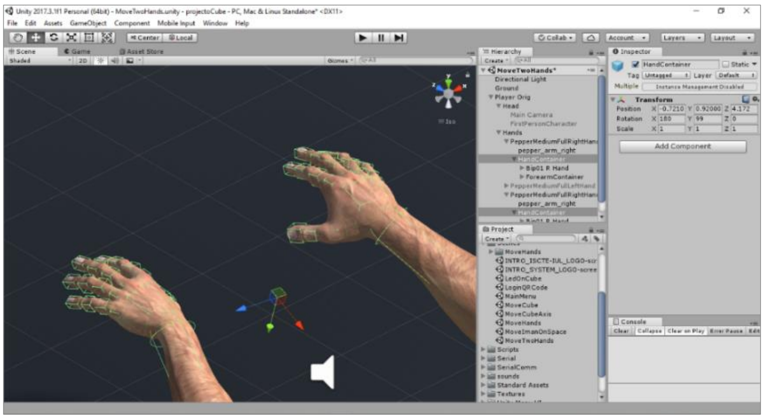

Figure 4.6 - Set of collider components attached on hands texture objects.

The goal of making physiotherapy exercises more accessible and highly interactive with the user, some Virtual Reality scenarios for the Gaming Platform was designed. Thus, the implemented system permits to animate an Avatar using a First-Person Controller. This controller method allows the game scene Camera to be placed in the position of the Avatar's eyes.

On Classic PC Games that use First-Person-Controller, this player construction process is usually used with the mouse cursor, where the rotation inputs of the Avatar are generated by the cursor movements. To avoid using the mouse cursor during the game, we will use an electric circuit like the one used in the smart gloves, which will not include analog sensors, just an IMU. The smart headband will be responsible for collecting the head rotation value of the patient, which will be used as the rotation input of the player's avatar head, that is the scene camera. Figure 4.7 shows the scene camera placed on the avatar's head in the game.

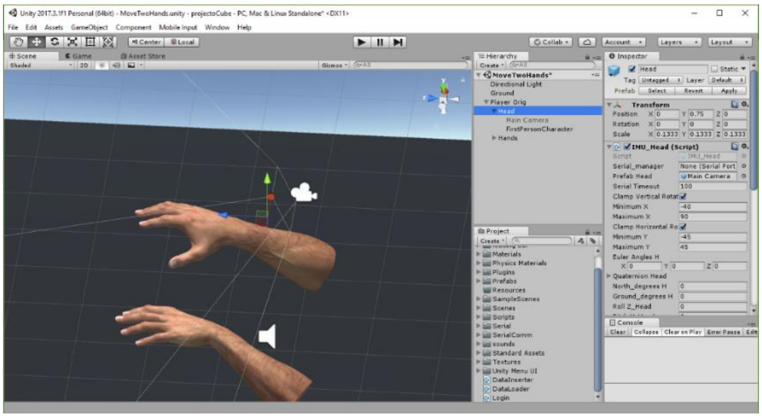

Figure 4.7 - The scene camera placed on the avatar's head in the game scene.

The process of serial ports reading within the game software is done using threads, as the system reads data from 3 different devices (2 devices of the smart gloves and 1 device mounted on the head) at the same time, allowing a fluid interaction with the virtual environment.

4.3 Developed Games

Cans Down and **Coffee Pong** games were developed for application. There are components that are similar in both games. The first concerns a countdown so the patient has time to prepare the hands and headband to make IMUs calibration process in front of monitor and to start playing. This time consists of 10 seconds for the patient to place their hands in front of monitor, the application will detect position and repositioning hands on the scene to start catch their movements. After this time has elapsed and the playing time starts, the game engine can replicate the hands' moves and the patient is operational to grab and drop virtual objects.

In the upper left corner are the patient's various scores, like number of objects picked, the number of objects throw and the number objects that was fall and in the center the clock with the remaining time for the game as illustrated in Figure 4.8. On lower corners we can observe the Force Feedback bars, that will show the values obtained from FlexiForce sensors by the Arduino microcontroller, as can be seen in Figure 4.9. Firstly, will explain the operation of the **CansDown** and subsequently the **CoffeePong**.

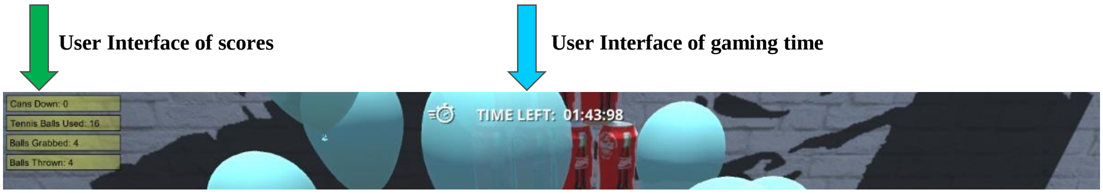

Figure 4.8 - The top graphic components of game scenes.

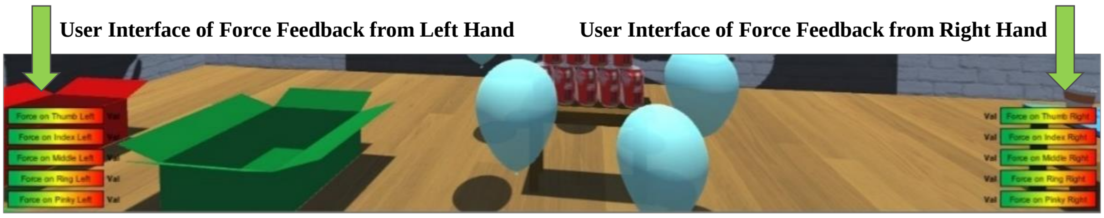

Figure 4.9 - The bottom graphic components of games.

4.3.1 Cans Down

The game goal is to knock down the stack of cans by throwing the tennis balls that appear in the golden goblets. To grab a ball, 2 fingers of one hand should touch it simultaneously, it should be the thumb and the other Finger of the hand. To throw the ball, the hand must be open at the end of the movement. A new ball is automatically generated whenever other ball touches the ground. Balloons are a decorative element of the Virtual Reality scenario. For each Can dropped on the floor is gained 1 point. Figure 4.9 illustrates the **Cans Down** game interface.

Figure 4.10 - The Cans Down game interface.

The graphical interface of this game includes a stack of cans arranged on a table in front of the player's Avatar start position in the scene. A set of balloons rotates around the stack, operating as a distracting element. The tennis balls objects emerge on the golden cups placed in the scene below the place where the upper limbs begin.

When the patient finishes the game, the results in terms of scores are stored in the database as well the data collected from wearable devices, described later in this chapter. Three different types of **scores** are stored:

- Total of tennis balls that have been grabbed;
- Total of tennis balls that have been thrown;
- Game score about cans dropped;

4.3.2 Coffee Pong

This game aims to exercise the upper limbs by capturing finer movements. It allows the patient to grab the available ball on the ping-pong table with one hand and pick up the ping-pong paddle with another. The goal is make balls take down coffee cups in front, on another side of table. For each ball thrown, a new ball is placed on table. If the patient drops the paddle to the floor, a new paddle is placed on start position. For each Coffee Cup dropped the patient gains one point. A Figure 4.11 shows an example of the **Coffee Pong** game interface.

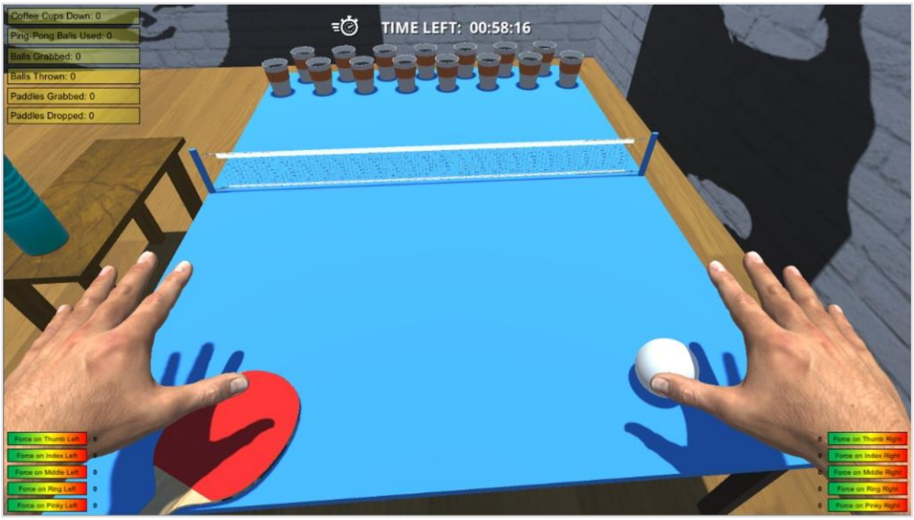

Figure 4.11 - The Coffee Pong game interface.

The graphical interface of this game includes a set of coffee cups arranged on the opposite side of a ping-pong table placed in front of the player's Avatar start position in the scene. The objects ping-pong balls and paddles emerge on the player's side table below the place where the upper limbs begin.

When the patient finishes the game, the results in terms of scores are stored in the database as well the data collected from wearable devices, described later in this chapter. Three different types of **scores** are stored:

- Total of ping-pong balls that have been picked;
- Total of ping-pong balls that have been thrown;
- Game score about coffee cups dropped;

4.4 Adaptability

Some considerations related to adaptability were included in the development of serious games. One of the points to be mentioned is the adaptability to the patient's gender, where for a male patient, the scene initiates the avatar of the player with a kit of masculine textures, and if the patient is female the textures to load refer to male / female hands (see Figure 4.12).

Figure 4.12 - Physio Wear Unity Application Game Scene on Cans Down Game, with female hands textures.

Another configuration point refers to the adaptability of the configuration of the limbs to be used during the exercises. Patients can perform the exercises with the right hand, left hand or both hands, and the game interface adapts to this configuration parameter, displaying only the objects of the selected members.

Another point of personalization refers to the difficulty of playing the game itself, which can fall into three levels for both games: EASY, MEDIUM and HARD. This parameter was included and thought during the development of the project. Its implementation was carried out on the side of the mobile application and the Web API, however due to the lack of time it was not implemented in the game. It is a parameter included in the settings of the exercise plan but still does not present any action on the scenario of virtual reality of serious games developed.

All these parameters are configurable in the creation of each exercise plan by the responsible physiotherapist.

4.5 Game Feedback

The feedback of the game is one of the most important components to encourage the patient to continue playing the game. In any game whether it is video game or a serious game, it must have a mechanism to respond to the actions performed by the user, for example, whenever the user correctly performs a goal a feedback must be given.

The feedback can be divided into two aspects, visual and auditory. With respect to the visual aspect this can be used to show a message indicating that you have performed a sequence of objectives or even to indicate how many points the patient won or lost by performing a certain task.

As for the auditory aspect, this can be used as background music, or in the case of performing a positive or negative task, for example, releasing a sound clip that conveys a sense of success in the case of a positive task, or a clip sound that conveys a sense of failure in the case of an unfinished or failed task.

Available application games use feedbacks, both visual and auditory. In the case of the Cans Down game, auditory feedback is used when the patient throws a ball to stack of cans and the background music is used during play so as not to make the game monotonous if the patient cannot hit the cans with ball. Visual feedbacks are used in several situations:

- To indicate whether the patient gained points hitting the cans with balls;
- To indicate the patient how many balls are picked up and throwed.
- The timer present in all games informs the remaining time.

In Both games have visual feedback about Force Feedback of each finger for each hand, like described above and illustrated in Figure 4.9.

A method that performs a displacement estimation based on the linear acceleration readings has also been developed. The graphical interface of this estimation is not displayed at the beginning of the Game scene but can be accessed by pressing the U key.

In case the user finishes the objectives of the games before the expected time of the session, a message is displayed informing the user, and the scene ends. This happens when in the Cans Down game all cans are knocked over or in the Coffee Cups Game, all cups are dropped.

4.6 Unity Application

The Unity application is expressed by a set of serious games for patients with hand and finger problems to aid in the rehabilitation process. After authentication in the application, data is collected to configure the games according to the characteristics of the patient, previously configured by the physiotherapist in mobile application.

To run the Unity application, a Bluetooth serial connection must already exist between the three smart devices. This setting is only made the first time the application is run so the operating system can configure the serial COM ports it will use to establish the connection to the devices when starting the game scenarios.

Each of the Bluetooth Devices has already been configured [see *attachments* - Technical Manual] during circuit development. The communication speed was set at 57600 baud, the names of the three devices are: HC-05-LEFT, HC-05-RIGHT and HC-05-HEAD. And the connection passwords for the 3 devices is the default: 1234. This procedure is performed in the Bluetooth settings of the operating system itself. After this process is performed for each device, two serial COM ports will be assigned to each port, one port for incoming communication and one for outcoming communication.

When initiating communication with the devices, Unity will look for the COM port of the output communication associated with the Name of each of the paired devices, allowing the communication with them. The LED state of the Bluetooth module (Red LED), stops blinking continuously and starts to do so at intervals. The Figure 4.13 shows Unity Application diagram.

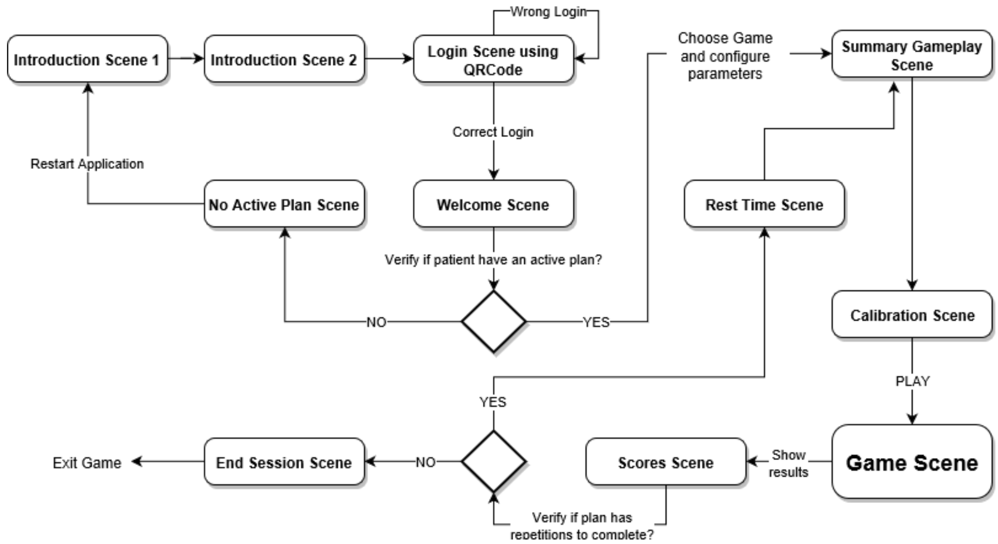

Figure 4.13 - Unity Application Diagram.

The diagram shows the logical sequence of all stages of the game application.

Following is presented the sequence of screens according diagram that are shown to users of the game application as well a description about them.

The first Introduction Scene features only an ISCTE-IUL logo. Then the second Introduction Scene is loaded. This includes the Physio Wear application logo, ISCTE-IUL and IT-IUL. (See Figure 4.14)

Figure 4.14 - Physio Wear Unity Application Introduction Scene 1 and 2.

Then the LoginQRCode Scene is loaded. This screen includes a first welcome message for the user, an access point to the Webcam of the computer where the application runs and an informative message reminding the user of the need to connect the Bluetooth of the computer (see Figure 4.15).

To log in to the Game Unity application, the patient must present the QR Code associated with his profile to the Webcam of the computer where the game is hosted. This can be given by the physiotherapist in paper format after the creation of your profile on the platform in the first consultation, or whenever it is necessary to be consulted on the profile page of each patient in the mobile application. If Login is wrong a message is shown to user and is played an audio clip with these feedback indication (see Figure 4.15).

When system's webcam detects a valid QR Code, the login is performed with credentials of the patient, a welcome message with name of patient is presented and is played an audio clip with these feedback indication (see Figure 4.15).

Figure 4.15 - Physio Wear Unity Application in LoginQRCode Scene, showing first welcome message, showing message of wrong Login and showing message of correct Login.

After login, a new screen is loaded - Welcome Scene - with some patient information such as name, gender and age (see Figure 4.16), while the system checks for an active exercise plan for the patient.

Figure 4.16 - Physio Wear Unity Application Welcome Scene.

If there is no exercise plan valid for that day, a new screen is displayed to the patient - No Active Plan Scene - showing an informative message to the patient (see Figure 4.17) and restarting the application.

Figure 4.17 - Physio Wear Unity Application No Active Plans Scene.

If there is a valid exercise plan for the day, the active plan information for that day will be uploaded and a new screen will appear with the summary of the exercise plan and serious game information, Summary Gameplay Scene (see Figure 4.18). Each training plan set up by the physiotherapist allows:

Figure 4.18 - Physio Wear Unity Application Summary Gameplay Scene.

- choose the game to play;
- the start and end dates for which the plan is valid;
- the duration of each session;
- the rest time between sessions;
- the number of repetitions of that plane;
- the members to use;
- the difficulty of the game;
- and some necessary notes;

It should be during the display of the last screen that the patient should, with help of someone, wear the smart gloves as well as fix the headband in the most correct place of the head.

After the set display time screen has elapsed, a new screen is loaded with calibration instructions and an example image for wearable devices, the Calibration Scene (see Figure 4.19). The Calibration mode is active on 10 first seconds of the Game Scene and can be activated by pressing the space bar off the computer whenever the user of the serious game so requires. The screen display time has been set at 90 seconds to allow the user to absorb all the information contained therein. The Game scene starts on when the timer of the display on this screen ends. The Game Scene is loaded in device calibration mode that ends after finishing the 10-second count displayed on the screen (see Figure 4.19). This step is fundamental to the game can be performed because it will allow the system to perceive the location of the monitor where the game is running and the position where the user will execute the movements.

After these 10 seconds of counting the game is started (see Figure 4.20). The game timer starts, and a new session is created for that set exercise plan. At the data level, a directory is created in the folder where the game is allocated, and files are created where all data is hosted locally in .csv format. About the collection of this data to the server, for

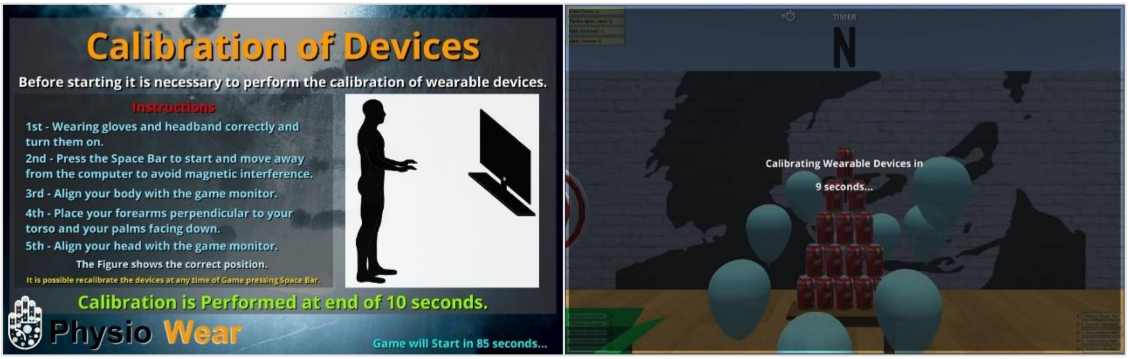

Figure 4.19 - Physio Wear Unity Application Calibration Scene and Application Game Scene on Calibration Mode on Cans Down Game.

each cycle the data collected from the devices are stored, a block with 200 readings is made, which when filled is sent to the server.

Figure 4.20 - Physio Wear Unity Application Game Scene on Cans Down Game.

A value of 200 readings was established to send the data acquired from the sensors of the wearable devices to be consistent with the use of the resources in the accesses to the system database. The issue is not to overload the server connection with too large data allocation requests, with a large amount of data, but also not to make too many requests during the execution of the game. This time, tests were performed with 50 readings, 100 readings and 200 readings with each of the devices. It was concluded that the time needed to make a new request after completing the previous one, in the internet connections available during the development, was quite acceptable and consistent with a 200 of readings against the lower values, on the other hand, it was achieved in proportion, the number of data allocation requests. It is concluded that this process in the future will require more dedication to better management compression in accessing system resources.

After completing each set of exercises, the patient is presented with a board of their scores and the session ends (see Figure 4.21).

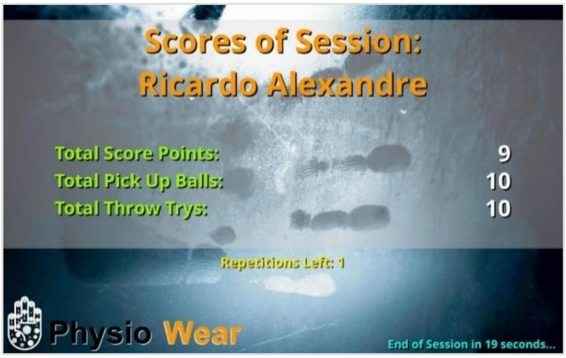

Figure 4.21 - Physio Wear Unity Application Scores Scene.

If the patient has repetitions to do regarding the exercise plan, the Rest Time Scene, which contemplates a loading bar attached with the rest time defined for the exercise plan, is loaded (see Figure 4.22). After this timeout, the Summary Plan Scene is loaded again. If the patient does not have repetitions to do regarding the exercise plan, the End Session Scene is loaded, which includes a new informative message to the patient (see Figure 4.22). After the screen display time finishes the application terminates.

Figure 4.22 - Physio Wear Unity Application Rest Time Scene and End Session Scene.

This covers all the main features of the Unity application. All images were captured in the built-in application PhysioWearUnity3D.exe created in Unity 2017.3.1f1 (64-bit) and running on Windows 10 Home Edition.

4.7 Data Acquisition

During rehabilitation sessions, data are collected and stored to create an Electronic Health Record, so they are accessible through the mobile application, serving as future analysis for physiotherapists.

After being logged in, and after receiving the active exercise plan, the game scene for each exercise plan is loaded. After the connection is established with the three devices, the session is created and the collection (through a parallel process independent of the reading thread) of the data received from the three devices during the execution of the exercises and recording in the database is initiated.

This process starts after ensuring that the current session was created for the loaded plan. For each of the devices a set of readings is created, defined in <u>200 readings</u>, which each time this set is filled is sent to the database. At the end of each session, the scores for each game are also sent to the database.

For each hand, the collected measured data are described below:

- The *rotation* (Euler Angles) values and *linear accelerations* of the pulses (forearm) are obtained and validated by the IMU module. The values are stored in degrees (-180º to 180º);
- The FlexSensors 2.2" capture the *angular flexion and compression* values exerted by each finger on each hand. The values are stored in degrees (0 to 90º) of bending according previous calibration in Embedded Software (ARDUINO Platform).
- The FlexiForce Sensors A201 capture the *pressure force* values exerted by each fingertip on each hand. The values are stored in percentage of measured force according previous calibration in Embedded Software (ARDUINO Platform) where 0 % corresponds to no readings measured and 100 % to maximum value measured.

The measured collected data associated with head are:

- The *rotation* (Euler Angles) values and *linear accelerations* are obtained and validated by the IMU module (-180º to 180º).

After the analysis it was necessary to understand where the information resulting from the execution of the training would be stored. The first hypothesis would be to create tables in the database to store the information, while the second hypothesis would be to create a file with a certain structure and save that file on the server. The second hypothesis was chosen due to the following factors:

- Due to the large amount of data produced by the games, the database would be under loaded with information, which could affect the rest of the system related search tasks;
- Saving the data to files that are stored on the server makes it possible to download the file for analysis in another program, for example, using MATLAB or Microsoft Excel.

The data is stored in a .CSV format to facilitate future analysis and exported to a folder in Game directory. It was created one file for each hand (LEFT and RIGHT) and other to the HEAD. Specific organization of the data was considered and following described. From each hand, system stores the following data:

- Time;
- Rotation Roll;
- Rotation Pitch;
- Rotation Yaw;
- Linear Acceleration X-axis;
- Linear Acceleration Y-axis;
- Linear Acceleration Z-axis;
- Flexi Force reading of pinkie finger;
- Flexi Force reading of ring finger;
- Flexi Force reading of middle finger;
- Flexi Force reading of index finger;
- Flexi Force reading of thumb finger;
- Flex Sensor reading of pinkie finger;
- Flex Sensor reading of ring finger;
- Flex Sensor reading of middle finger;
- Flex Sensor reading of index finger;
- Flex Sensor reading of thumb finger;

From the head, system stores:

- Time;
- Rotation Roll;
- Rotation Pitch;
- Rotation Yaw;
- Linear Acceleration X-axis;
- Linear Acceleration Y-axis;
- Linear Acceleration Z-axis;

The data to be stored in independent files provides an extra analysis tool so that physiotherapists can perform other types of analysis through the mobile application. Figure 4.23 shows an example of the file structure for the Right Hand when it is opened by the Microsoft Excel program.

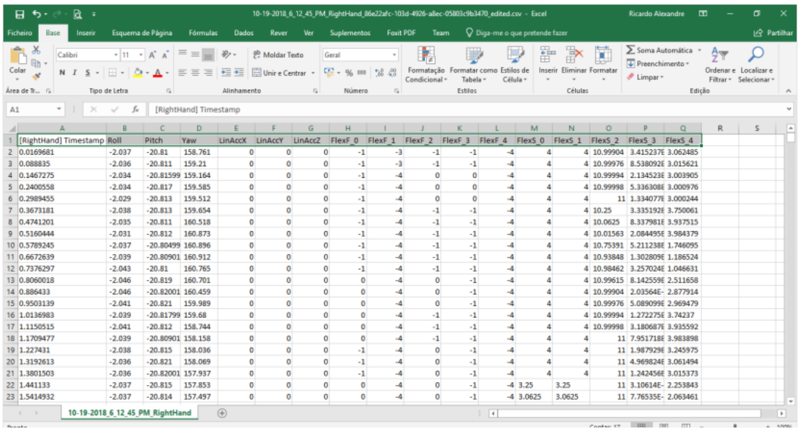

Figure 4.23 - Example of data from a file with coordinates referring to the patient's hands and fingers on Microsoft Excel program.

Chapter 5 - Server Side

Overview

This chapter is intended to describe Physio Wear's Server Side and how communications are performed between the server and the other applications (Unity 3D and Mobile App).

5.1 Server

The Server is responsible for storing all the information regarding clinic patients, training and results, and the main objective is that all information is centralized in one place and can be accessed remotely by the system applications. By storing this data, it is possible to analyse and see the information in the future, either with the mobile application destined for the manager, physiotherapists and patients, but also for Unity 3D, so the application can collect information and configure the games designed for the patient.

To enable the exchange of information between the Physio Wear applications and the server, a Web Services called Representational State Transfer (REST) is used. Web Servers are computers responsible for accepting HTTP requests from clients and serving them with HTTP responses. The process starts with a connection between the terminal, where the Web Server is installed and the client terminal, for example, a browser or a mobile application, and the Web Server must be always available and operational, because it is not possible to predict when a connection will be made between the two points. After the connection is made, the customer's request is processed (based on the security restrictions defined and requested information) and a response is returned from the server. These Web Servers can also run programs or scripts to interact with the client. Content sent by the Web Server in response to an HTTP request can have two types:

- Static Pages: the content comes directly from an existing file on the server, that is, regardless of which client accessing the information will always be the same;

- Dynamic Pages: the content is dynamically created by another script program or API called by the server. The request, once received, is processed by the Web Server that will dynamically create the content that will then be sent to the client.

Dynamic pages have the advantage that they can be programmed using some programming language such as PHP, JAVA, C# and others. Through these it is possible to program them to access a DB and present personalized information, which is sent to the client.

The Physio Wear system uses this type of API to process the customer's request, access the DB and send the information, through C# scripts of a Web API created with ASP .NET Core v2.1 [77] using Entity Framework Core [78] to access a MySQL Database (version mysql-5.7.11).

All components of this architecture are freeware and open-source, only the server was being provided by the dissertation advisor. The server is running on a remote Windows machine.

5.2 Database

The Database (DB) stores all of the data used and created by the applications, making it one of the main components of the Physio Wear system, and its design and implementation was done to ensure the correct operation of the system.

The entire database structure was developed in the MySQL Workbench v6.3 program [79]. This program has several tools to create a database through its visual presentation. The database was developed using MySQL v5.7.11. This Database Management System (DBMS) was chosen because it was the one that best suited the development needs of the project. Initially a system needs analysis was performed to find out what kind of data the database should store. After completion, the database was exported to a file, using MySQL Workbench, to later import that file to the server through the phpMyAdmin tool [80]. In Figure 5.1 it is possible to observe the final diagram of the database.

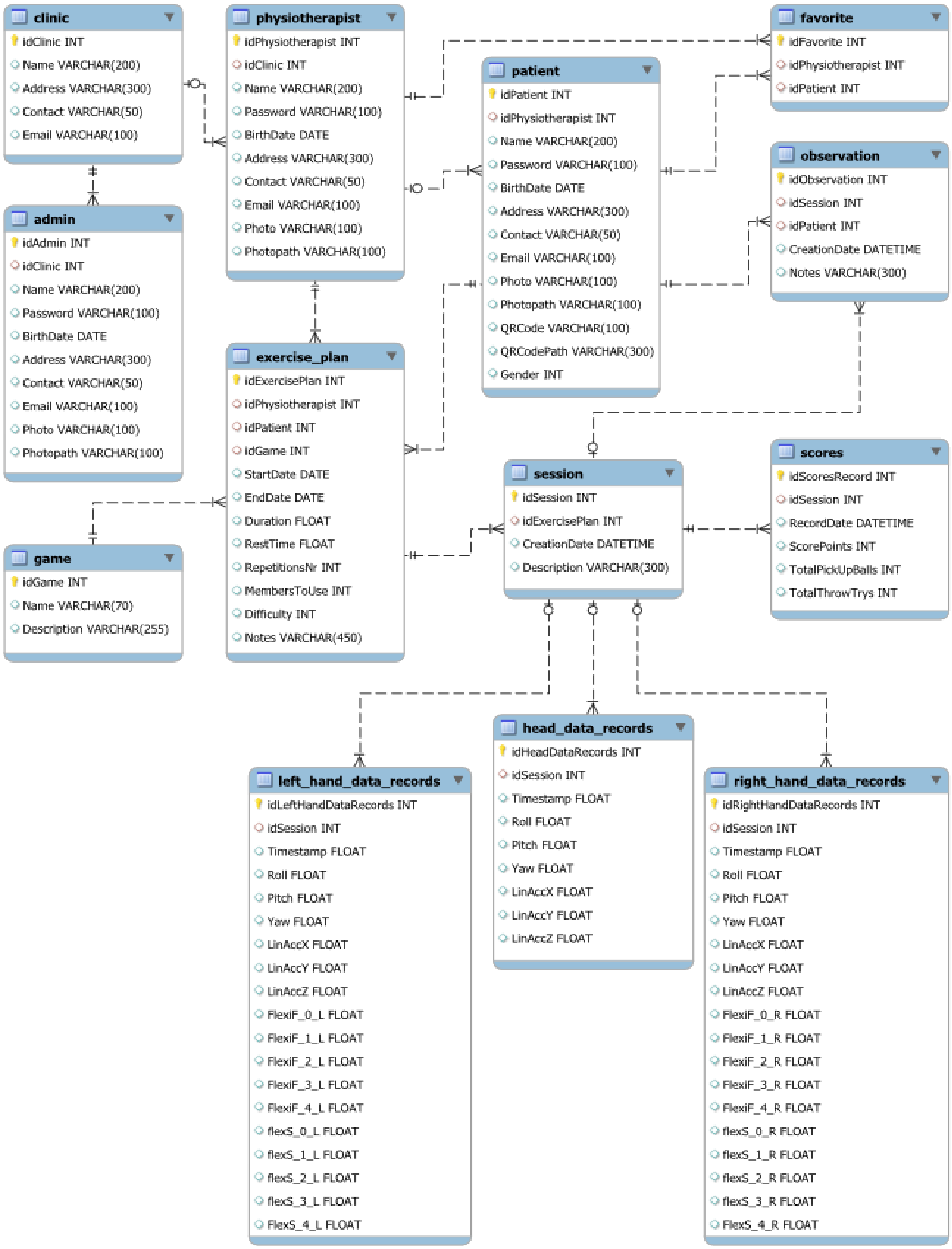

Figure 5.1 - Diagram Database.

The database consists of **thirteen** tables:

- **clinic**: this table stores the data related with the physiotherapy clinic. Fields such as the name of the clinic, address, contact and e-mail are present in this table. The primary key is idClinic.

- **admin**: stores the data for the clinic manager. It is in this table that the credentials (valid e-mail and password) are present for the manager to enter in the mobile application. Fields such as name, address and contact are present in this table. The table it is prepared to store the name of the manager's photo, through the Photo field, which is stored on the server. The Photopath field indicates the path to the folder where the manager's photo is hosted in relation to mobile app. Contains a foreign key (idClinic) that matches the clinical table, thus indicating which clinic the manager belongs to. The primary key is idAdmin.
- **physiotherapist**: stores the data concerning the physiotherapists of the clinic. It is in this table that the credentials (valid e-mail and password) are present for the physiotherapist to enter the mobile application. Fields such as name, address, date of birth and contact are present in this table. The table it is prepared to store the name of the physiotherapist's photograph, through the Photo field, which is stored on the server. The Photopath field indicates the path to the folder where the photograph is hosted in relation to mobile app. Contains a foreign key (idClinic) that matches the clinical table, thus indicating which clinic the physiotherapist belongs to. The primary key is idPhysiotherapist.
- **patient**: stores the data for patients in the clinic. It is in this table that the patient's credentials are present (valid e-mail and password) for them to enter in the mobile application. To access the Unity 3D App, they have a QRCode. This field is automatically generated by the mobile application and this is the content that is inserted in the QRCode field of the table, being an essential field to check the authentication. Fields such as the name, identification number, gender, address, contact, and e-mail are present in this table. The Photo and QRCode fields indicate the name of the photograph of each component that is stored on the server. The Photopath and QRCodePath fields indicate the path where the photograph is hosted in relation to mobile app. The gender field indicates the gender of patient. Contains a foreign key (idPhysiotherapist) that matches the physiotherapist table, indicating which physiotherapist is responsible for that patient. The primary key is idPatient.
- **favorite**: each physiotherapist using the mobile application can mark a patient as a favorite. It is in this table that this information is stored. Contains two foreign keys, idPhysiotherapist and idPatient, which correspond to the physiotherapist and clinic table,

respectively, thus indicating that a patient is signalled as a favorite for a physiotherapist. The primary key is idFavorite.

- **exercise_plan**: it is in this table that the configurations relative to each exercise planning are stored, being an essential table for the patients to execute the training created by the physiotherapists. The StartDate field indicates the start date for which this workout is available. The EndDate field indicates the end date of the workout, for example, the date by which the patient can perform the training. The Duration field indicates the time the patient must play each game. The RestTime field indicates the rest interval between two games. The RepetitionNr field indicates how many times that game must be repeated. The MembersToUse field indicates which hand to use (LEFT, RIGHT or BOTH). The Difficulty field indicates the difficulty of the game (EASY, MEDIUM or HARD). The Notes field allows the physiotherapist responsible for the training to write some observation. This table contains three foreign keys, the idPatient to indicate to which patient this training belongs, the idPhysiotherapist that indicates which physiotherapist is responsible for the training and the idGame that indicates the game to play. The primary key is idExercisePlan.
- **game**: stores the data regarding the games. It is in this table that this information is stored through field Name and Description. The primary key is IdGame.
- **observation**: stores the data regarding the observations created by the physiotherapists for the patients. This table has a field with the creation date of the observation and its description. Contains two foreign keys (idPatient and idSession) that matches the patient table, thus indicating which patient belongs to the observation and from what session realized. The primary key is idObservation.
- **session**: stores the data regarding the sessions performed by patients for the exercise plans created by physiotherapists. This table has a field with the creation date of the session and its description. Contains one foreign key (idExecisePlan) that matches the exercise_plan table, thus indicating which exercise plan that belongs the session completed. The primary key is idSession.
- **scores**: this table is intended to store the results of the training performed by each patient on the games. The RecordDate field stores the date that the result was entered the table. The ScorePoints, TotalPickUpBalls and TotalThrowTrys fields report to patient scores. It has a foreign key that matches the session table, thus indicating that certain

results come from certain session, which exercise plan is performed by a patient. The primary key is idScoresRecord.

- **left_hand_data_records**: this table is intended to store the readings measured in the session for each patient when using the wearable devices, on this case for the left hand. The Timestamp field stores the time that the reading was entered in the table. The Roll, Pitch and Yaw fields store values about the rotations, the LinAccX, LinAccY and LinAccZ store values about the linear accelerations, all these measured by the IMU. The remaining fields are to store both the Flexi Force and Flex Sensor readings, all registered entries follow the order, from the little finger to the thumb. It contains a foreign key that matches the session table, thus indicating that certain measured readings come from certain session, which exercise plan is performed by a patient. The primary key is idLeftHandDataRecords.
- **right_hand_data_records**: this table is intended to store the readings measured in the session for each patient when using the wearable devices, on this case for the right hand. The Timestamp field stores the time that the reading was entered in the table. The Roll, Pitch and Yaw fields store values about the rotations, the LinAccX, LinAccY and LinAccZ store values about the linear accelerations, all these measured by the IMU. The remaining fields are to store both the Flexi Force and Flex Sensor readings, all registered entries follow the order, from the little finger to the thumb. It contains a foreign key that matches the session table, thus indicating that certain measured readings come from certain session, which exercise plan is performed by a patient. The primary key is idRightHandDataRecords.
- **head_data_records**: this table is intended to store the readings measured in the session for each patient when using the wearable devices, on this case for head. The Timestamp field stores the time that the reading was entered in the table. The Roll, Pitch and Yaw fields store values about the rotations, the LinAccX, LinAccY and LinAccZ store values about the linear accelerations, all these measured by the IMU. It contains a foreign key that matches the session table, thus indicating that certain measured readings come from certain session, which exercise plan is performed by a patient. The primary key is idHeadDataRecords.

The Photo and Photopath fields in the corresponding user tables of the system (admin, physiotherapist and patient) were not implemented in the mobile application due to time constraints and because they were non-fundamental components in the evaluation of the system. It is envisaged that they will be implemented in the future in order to make the profiles of users of the system even more interactive and reliable by presenting a photograph of the users.

5.3 Web API

The use of a Web API is required to perform communication between the server and the applications (Unity and Mobile Apps). A Web API is an application programming interface (API) for both a server and a client. It is used to recover only the required value in a database. These API are stored on the server, and when it is necessary to perform some search operations, change data or even erase data, the corresponding method is invoked using a HTTP (Hypertext Transfer Protocol) connection. The HTTP is designed to enable communications between clients and servers and acts as a request-response protocol between client and server (see Figure 5.2), for example, a client submits a HTTP request to the server and it returns a response to the client. This response contains information about the request and may also contain the requested content.

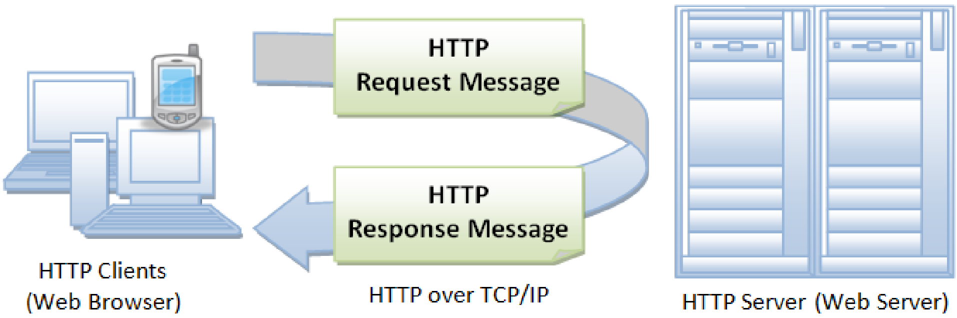

Figure 5.2 - HTTP Request-Response Diagram [79].

For each model object in our Web API, that is, for each table in our database, there are a set of methods that can be performed. These represent the actions that can be performed in the tables and are usually delimited in Controller classes. The most usual methods for a request-response between the client and the server are:

- **GET** requests request data from a specific resource.
- **POST** requests submit the data to be processed in a specific resource.
- **PUT** requests allow you to change the data for a specific resource.
- **DELETE** requests allow you to erase the data of a specific resource.

Following is presented a table that includes all the Controllers created for access to the database models, as well as the request-response methods implemented in the Web API.

Table 4 - The Controllers created for access to the database models, as well as the request-response methods implemented in the Web API.

LoginController	
POST: api/Login:	This post method was created to check if the user's credentials to access the system (email address and password) are present in the database. It returns a negative response if it does not find the credentials, and positive if it finds them. The method performs the verification for the 3 types of system users (Admin, Physiotherapist and Patient).
ClinicController	
GET: api/Clinic/PerPatient/{idPatient}:	This get method was created to return a clinic based on idPatient. It returns a negative response if it does not find it, and positive if it finds it.
GET: api/Clinic/{idClinic}:	This get method was created to return a clinic based on idClinic. It returns a negative response if it does not find it, and positive if it finds it.
POST: api/Clinic:	This post method was developed to create a clinic in system. It returns a negative response if creation model is invalid and positive when successful.
PUT: api/Clinic/{idClinic}:	This put method was developed to update the contents of a clinic in the system based on idClinic. It returns a negative response if model is invalid, or idClinic does not match, and positive when successfully changed.
DELETE: api/Clinic/{idClinic}:	This delete method was developed to remove a clinic from the system based on idClinic. It returns a negative response if it does not find it, and positive if it finds it and successfully removes it.

AdminController	
GET: api/Admin/{idAdmin}:	This get method was created to return the admin data based on idAdmin. It returns a negative response if it does not find it, and positive if it finds it.
POST: api/Admin:	This post method was developed to create an admin in the system. It returns a negative response if creation model is invalid and positive when successfully.
PUT: api/Admin/{idAdmin}:	This put method was developed to change the contents of an admin in the system based on idAdmin. It returns a negative response if model is invalid, or idAdmin does not match, and positive when successfully changed.
DELETE: api/Admin/{idAdmin}:	This delete method was developed to remove an admin from the system based on idAdmin. It returns a negative response if it does not find it, and positive if it finds it and successfully removes it.
PhysiotherapistController	
GET: api/Physiotherapist/ PerPatient/{idPatient}:	This get method was created to return a physiotherapist based on idPatient. It returns a negative response if it does not find it and positive if it finds it.
GET: api/Physiotherapist/ ForClinic/{idClinic}:	This get method was created to return a list of all physiotherapists of a clinic based on idClinic.
GET: api/Physiotherapist/ {idPhysiotherapist}:	This get method was created to return a physiotherapist based on idPhysiotherapist. It returns a negative response if it does not find it, and positive if it finds it.
POST: api/Physiotherapist:	This post method was developed to create a physiotherapist in system. It returns a negative response if creation model is invalid and positive when successfully created.

PUT: api/Physiotherapist/ {idPhysiotherapist}:	This put method was developed to update the contents of a physiotherapist in the system based on idPhysiotherapist. It returns a negative response if model is invalid, or idPhysiotherapist does not match and positive when successfully changed.
DELETE: api/Physiotherapist/ {idPhysiotherapist}:	This delete method was developed to remove a physiotherapist from system based on idPhysiotherapist. It returns a negative response if it does not find it, and positive if it finds it and successfully removed it.
PatientController	
GET: api/Patient/ ForPhysiotherapist/ {idPhysiotherapist}:	This get method was created to return a list of all patients of a physiotherapist based on idPhysiotherapist.
GET: api/Patient/{idPatient}:	This get method was created to return a patient based on idPatient. It returns a negative response if it does not find it, and positive if it finds it.
POST: api/Patient:	This post method was developed to create a patient in system. It returns a negative response if creation model is invalid and positive when successfully.
PUT: api/Patient/{idPatient}:	This put method was developed to update the contents of a patient in the system based on idPatient. It returns a negative response if model is invalid, or idPatient does not match and positive when successfully changed.
DELETE: api/Patient/{idPatient}:	This delete method was developed to remove a patient from system based on idPatient. It returns a negative response if it does not find it, and positive if it finds it and successfully removes it.
GameController	
GET: api/Game:	This get method was created to return a list of all games of the system.

FavoriteController	
GET: api/Favorite/ {idPatient}:	This get method was created to return if a Patient is favorite based on idPatient. It returns a negative response if it does not find it and positive if it finds it.
GET: api/Favorite/ ForPhysiotherapist/ {idPhysiotherapist}:	This get method was created to return a list of all favorite patients of a physiotherapist based on idPhysiotherapist.
POST: api/Favorite:	This post method was developed to create a favorite patient for physiotherapist in system. It returns a negative response if creation model is invalid and positive when successfully.
ExercisePlanController	
GET: api/ExercisePlan/ ForPatient/{idPatient}:	This get method was created to return a list of all exercise plans of a patient based on idPatient.
GET: api/ExercisePlan/ ValidsPerPatient/ {idPatient}:	This get method was created to return a list of valid exercise plans of a patient based on idPatient.
GET: api/ExercisePlan/ {idExercisePlan}:	This get method was created to return an exercise plan based on idExercisePlan. It returns a negative response if it does not find it and positive if it finds it.
PUT: api/ExercisePlan/ {idExercisePlan}:	This put method was developed to change the contents of an exercise plan in system based on idExercisePlan. It returns a negative response if model is invalid or idExercisePlan does not match and positive when successfully changed.
POST: api/ExercisePlan:	This post method was developed to create an exercise plan system. It returns a negative response if creation model is invalid and positive when successfully.
DELETE: api/ExercisePlan/ idExercisePlan}:	This delete method was developed to remove an exercise plan from system based on idExercisePlan. It returns a negative response if it does not find it and positive if it finds it and successfully removed.

SessionController	
GET: api/Session/ ForExercisePlan/ {idExercisePlan}:	This get method was created to return a list of all sessions based on idExercisePlan.
GET: api/Session/ {idSession}:	This get method was created to return a session plan based on idSession. It returns a negative response if it does not find it and positive if it finds it.
POST: api/Session:	This post method was developed to create an session in system. It returns a negative response if creation model is invalid and positive when successfully.
ScoresController	
GET: api/Scores/ {idSession}:	This get method was created to return a list of all scores based on idSession.
POST: api/ Scores:	This post method was developed to create scores in system. It returns a negative response if creation model is invalid and positive when successfully.
ObservationController	
GET: api/Observation/ ForSession/{idSession}:	This get method was created to return an observation based on idSession. It returns a negative response if it does not find it and positive if it finds it.
GET: api/Observation/ {idObservation}:	This get method was created to return an observation based on idObservation. It returns a negative response if it does not find it and positive if it finds it.
POST: api/Observation:	This post method was developed to create an observation in system. It returns a negative response if creation model is invalid and positive when successfully.
PUT: api/Observation/ {idObservation}:	This put method was developed to change the contents of an observation in system based on idObservation. It returns a negative response if model is invalid or idObservation does not match and positive when successfully changed.

LeftHandDataRecordsController	
GET: api/ LeftHandDataRecords/ {idSession}:	This get method was created to return a list of all LeftHandDataRecords based on idSession. It returns a negative response if it does not find it and positive if it finds it.
POST: api/ LeftHandDataRecords:	This post method was developed to create LeftHandDataRecords in system. It returns a negative response if creation model is invalid and positive when successfully.
RightHandDataRecordsController	
GET: api/ RightHandDataRecords/ {idSession}:	This get method was created to return a list of all RightHandDataRecords based on idSession. It returns a negative response if it does not find it and positive if it finds it.
POST: api/ RightHandDataRecords:	This post method was developed to create RightHandDataRecords in system. It returns a negative response if creation model is invalid and positive when successfully.
HeadDataRecordsController	
GET: api/HeadDataRecords/ {idSession}:	This get method was created to return a list of all HeadDataRecords based on idSession. It returns a negative response if it does not find it and positive if it finds it.
POST: api/HeadDataRecords:	This post method was developed to create HeadDataRecords in system. It returns a negative response if creation model is invalid and positive when successfully.

Chapter 6 - Mobile App

Overview

This chapter describes the mobile application of the Physio Wear System, how the initial evaluation took place, what are the main features in the application for users and how it was implemented and what resources it uses.

6.1 Initial Evaluation

The mobile application was developed as a working tool to allow the physiotherapist to manage all the information about the physiotherapy sessions, but also to allow future analyses on the results obtained by the patients with the full use of the system. The need to have a well-designed and intuitive application serves to help the physiotherapist and the patient by improving the communication between the two, about the evolutionary process of the patient.

Since the beginning of the project, that an important requirement for the system was the existence of a mobile application supporting the entire system. The reason for this is that the physiotherapist is always in constant movement to assist his patients, which in the case of an application on the computer would oblige the physiotherapist to move to a certain place where the software would be installed, drawing his/her attention away from the patients and causing some discomfort. The mobile application presents the advantages of the mobile device that can follow the physiotherapists. The same happens with the patients. For this project requirements, the doubt was which development platform should be used, with the alternatives considered (see Figure 6.1):

- Native application for iOS;
- Android native application for the Android operating system;
- Windows 10 devices applications;
- Cross platform application, developed in Xamarin, for Android, iOS and UWP.

Figure 6.1 – Platforms considered for mobile application development.

A native application can optimize the application for the platform that is intended for it by leveraging the features. However, one of the great advantages of having a cross-platform application is that it can be used on more than one platform using the same development code, for example iOS, Android and UWP. In the case of Xamarin, there is a fundamental and important detail that some cross-platform systems do not. This is the Xamarin natively compiles for each of the platforms, with the respective compilers, meaning that, compared to systems that create for example Apps with HTML pages and JavaScript, it confers a substantially higher performance and very close to the native.

Another key detail is that Xamarin, by default, uses the native controls of each platform, which generates applications, from the point of view of usability and user experience, "familiar" for the user and with the same type of use of the different controls. In the end, the choice fell on a cross-platform application for devices. The only functionality of the mobile application is to visualize data and make configurations, so the charts and some features may not be exactly the same in each of the platforms, but they will be similar and functional. This allows the application to run on any device such as a PC, Tablet or Mobile Phone, on any of the most commercial platforms on the market (Android, iOS and Windows). All the development resources are free and there is the possibility of developing native applications.

To run the application, the device must have access to the internet, either through a mobile network (3G, 4G) or Wi-Fi.

The application was developed using Microsoft Visual Studio Community 2017 software version 15.7.5 (see Figure 6.2). Visual Studio is the official IDE for Xamarin Forms applications development. For its development in terms of programming was used C#.

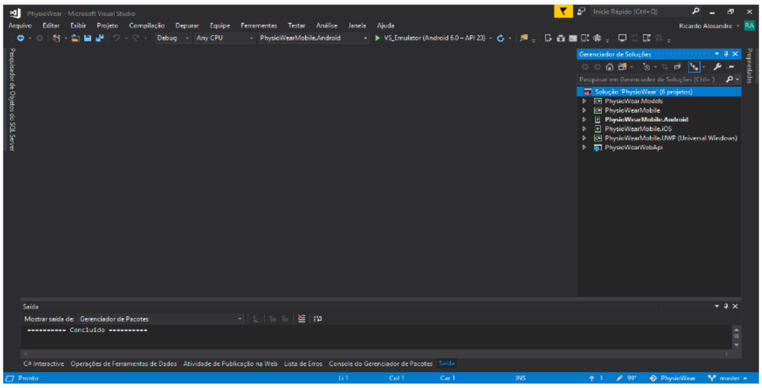

Figure 6.2 - Microsoft Visual Studio Community 2017.

6.2 Structure

Three types of user can use the mobile app:

- The **patient:** to access his QR Code of access to the game platform, to consult plans of exercises and to see results of the sessions;
- The **physiotherapist:** to perform queries on patient data, create exercise plans, register patients or visualize results of sessions.
- The **clinic manager:** to register physiotherapists in the system or view information about them.

It is designed to serve as an interface between the users and the system, providing the necessary resources for its use. Figure 6.3 illustrates the use cases that the users can perform in the application.

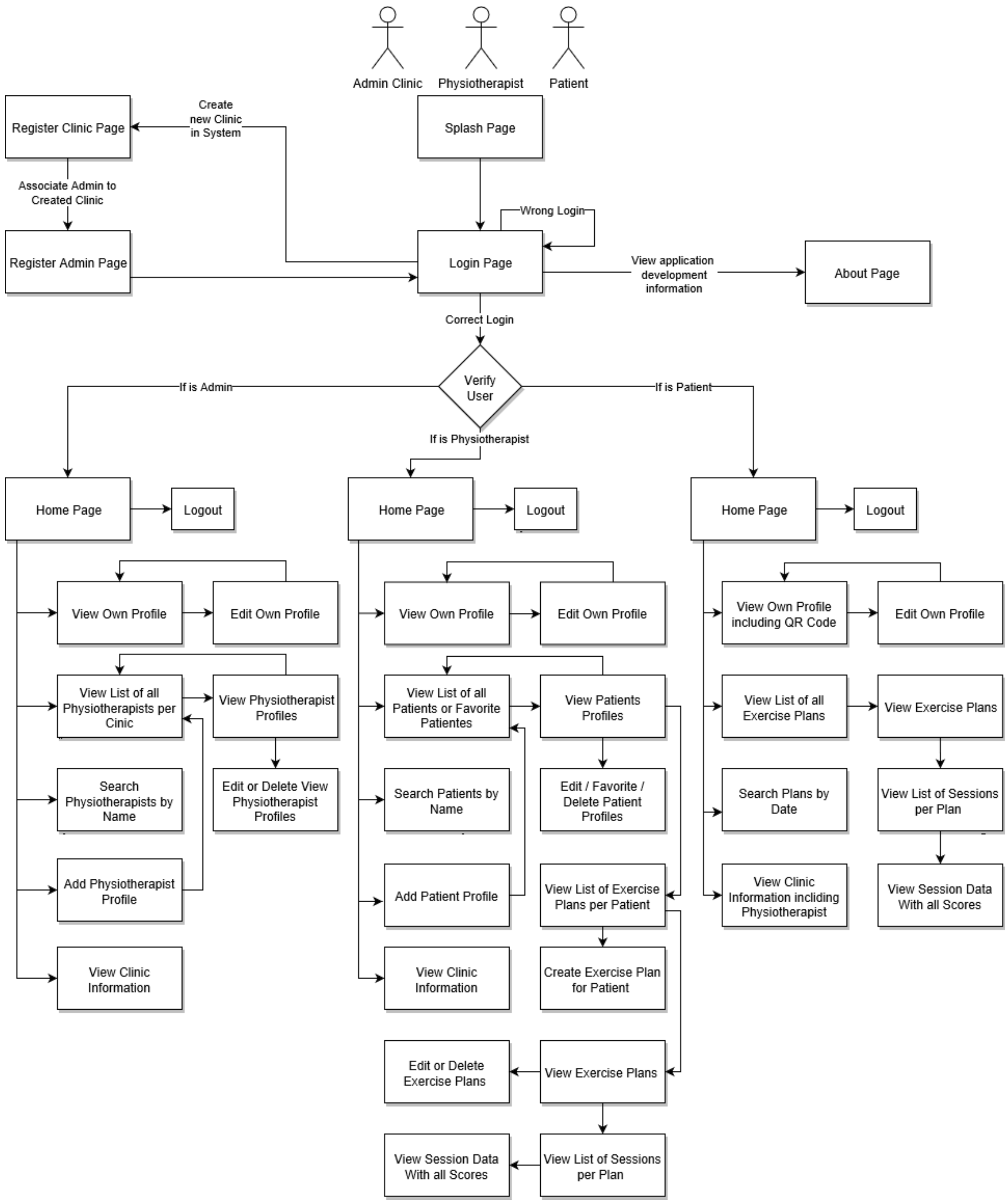

Figure 6.3 - Structure of the functionalities of the users in the mobile application.

6.3 Design and Implementation

The interface is one of the most important things to keep in mind when developing a mobile app. An application without a simple and intuitive interface becomes less attractive to its end users. Navigation must be consistent to create the best possible user experience.

For an application to have good user experience, the standards defined by Xamarin for the creation and development of the Graphical User Interface (GUI) have been followed and respected. The application was developed from the beginning, starting with the logo (see Figure 6.4).

Figure 6.4 - Physio Wear Mobile Application Logotype.

When the application starts, the splash page with logo animation is launched (see Figure 6.5). Then the login page is loaded. The user performs the authentication by entering their valid e-mail address and PIN in the corresponding text areas. In the following figure it's possible to verify the filled text areas (See Figure 6.6).

To access the Xamarin Application development information (see Figure 6.7), there is an About Page access button on the Login Page.

Figure 6.5 - Splash page.

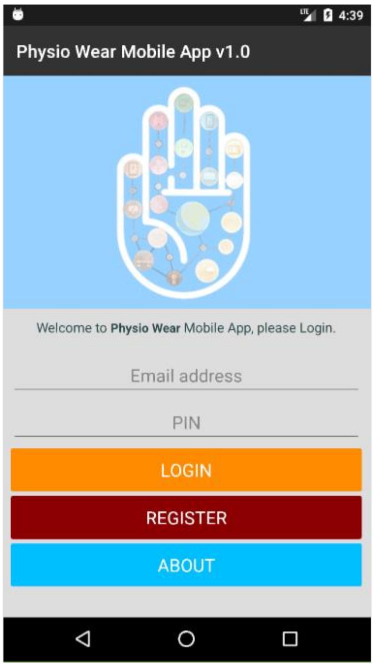

Figure 6.6 - Login page.

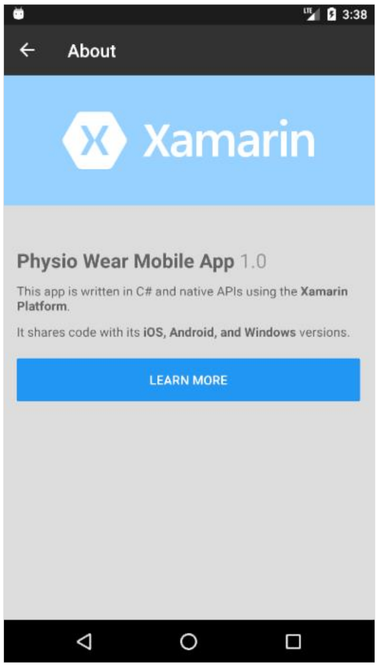

Figure 6.7 - About page.

The necessity of registration the corresponding clinics and administrators implied the creation of Clinical Registry and Administrator Registry features in two pages accessible on the Login Page. In the Clinic Registry page, it can fill in the information to register a new clinic (see Figure 6.8), then the registration page of the corresponding Administrator is loaded, where the registration form of the administrator of the clinic created will be filled out (see Figure 6.9).

These registries, for the moment, do not present any security mechanism, reason why any user can make the registration of clinics. These processes, in the future use of the system, should be corrected using for access to a super Administrator who will make these records in a secure way and preventing this process from being carried out by any user of the system. This feature was not implemented due to time constraints.

After the Login is successful, the Home Page appears. The application loads a Master Detail Menu, where the Master Menu page includes the page accesses: Home Page, My Profile Page, Add Profile Page, Clinic Information Page and the logout button (see Figure 6.10). The Detail page includes the Home Page (see figures below 6.11, 6.12 and 6.13).

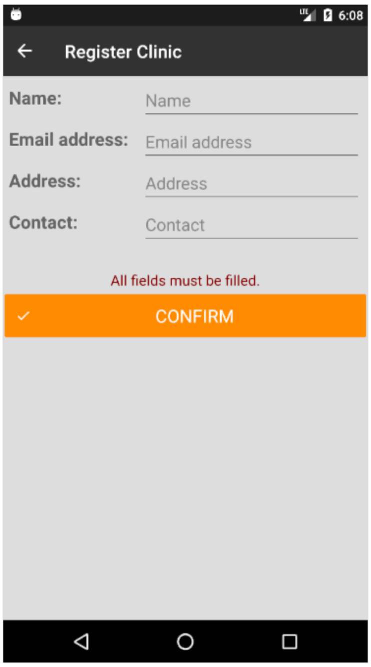

Figure 6.8 - Register Clinic page.

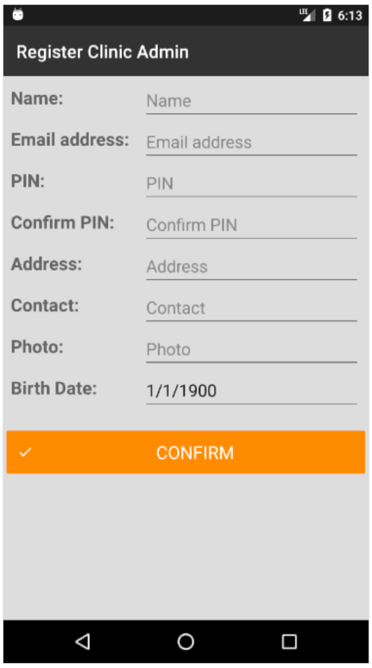

Figure 6.9 - Register Admin Clinic page.

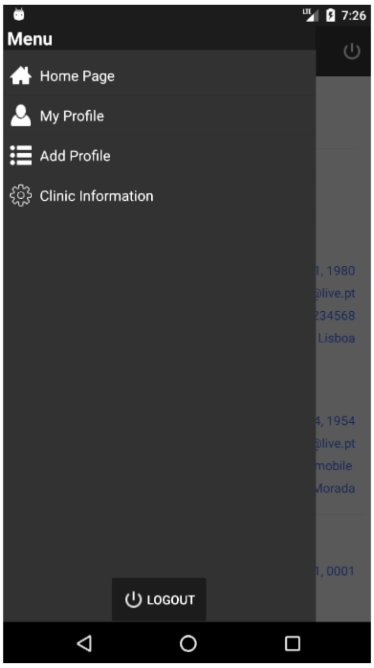

Figure 6.10 - Master Main Menu page.

In the case of the Admin login, the Home Page will be constituted by the list of Physiotherapists of the corresponding Clinic, a Search Bar and Add Profiles button (see Figure 6.11).

If the logged user is the Physiotherapist, the list will show the Patients associated with logged Physiotherapist and two plus buttons are adding to show the favourites Patients or all completed list (see Figure 6.12).

If the logged user is the Patient, the Home Page will be constituted by the list of Exercise Plans of the logged Patient (see Figure 6.13).

The list of profiles shown in both the Admin account and the Physiotherapist account will appear in alphabetical order. The list of exercise plans for the Patient account will appear sorted from the most recent to the oldest plan. This page includes a refresh list button and a logout button too.

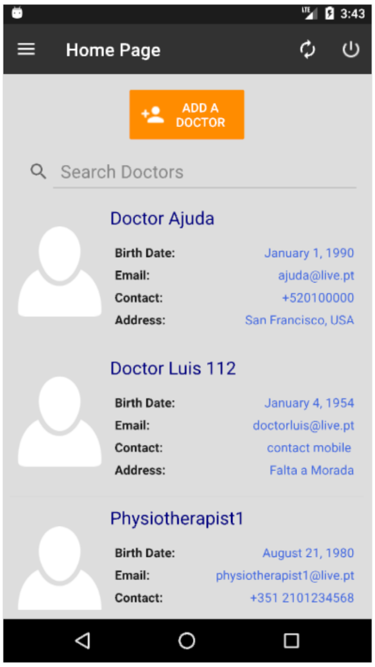

Figure 6.11 - Admin Home Page.

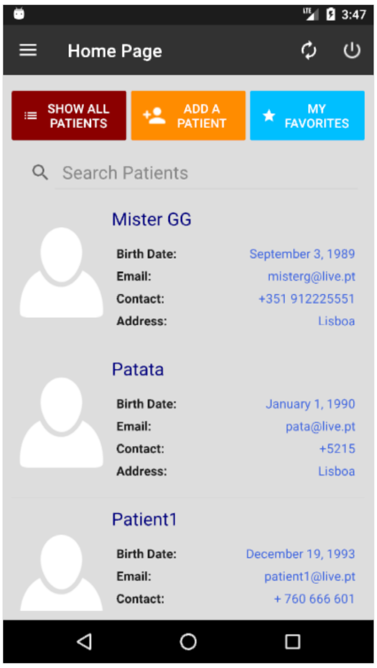

Figure 6.12 - Physiotherapist Home Page.

Figure 6.13 - Patient Home Page.

The My Profile page shows information about the user's own profile (see Figure 6.14). On My Profile Page of Patient there is also a button, which when pressed shows the QR Code associated with the patient's account, which will serve to login in the Game application (see Figure 6.15). It is possible to edit the information by clicking on Edit Profile Button (see Figure 6.16).

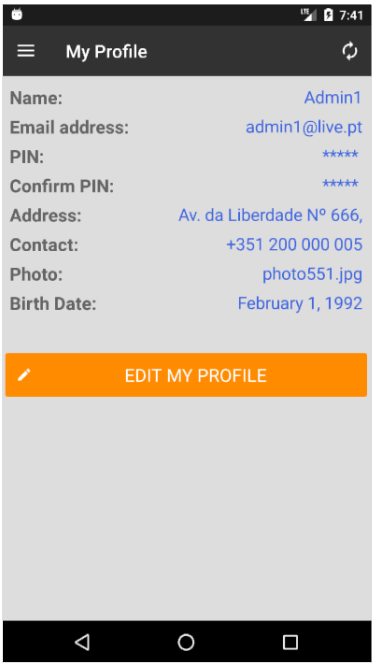

Figure 6.14 - My Profile page.

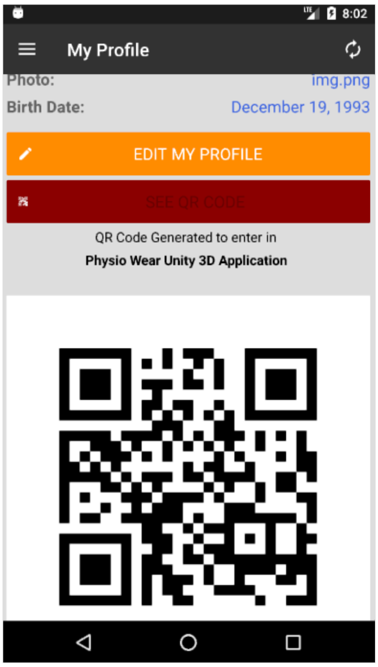

Figure 6.15 - My Profile Patient Account page showing QR Code.

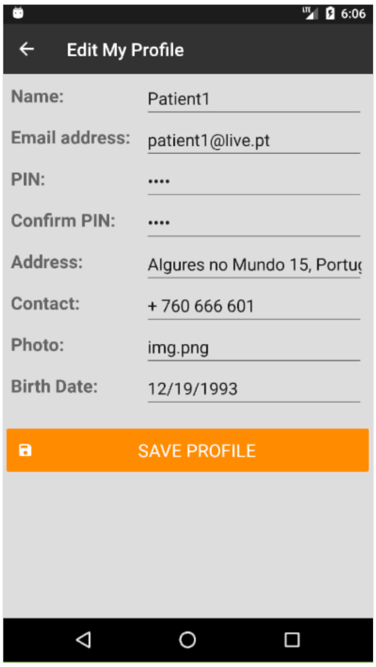

Figure 6.16 - Edit My Profile page.

The Clinic Information Page, shows the information Clinic for all users (see Figure 6.17). On Patient Account this page will show the information about the responsible Physiotherapist (see Figure 6.18)

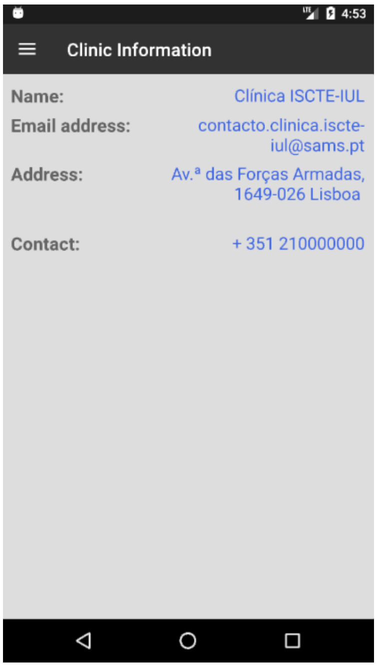

Figure 6.17 - Clinic Information page.

Figure 6.18 - Clinic Information on Patient Account page.

The Add Profile page allows the Admin to fill a form to create Physiotherapists for the clinic (see Figure 6.19), or in case of Physiotherapists allows to fill a form to create Patients of the specified physiotherapist (see Figure 6.20). The pages will be identical, the difference is the gender selection on Adding Patient Profiles.

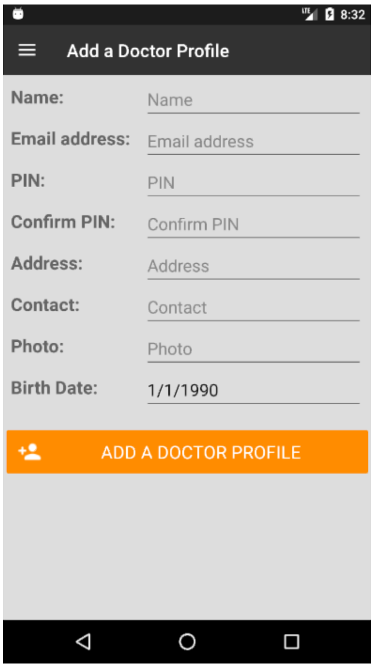

Figure 6.19 - Add Patient Profile page.

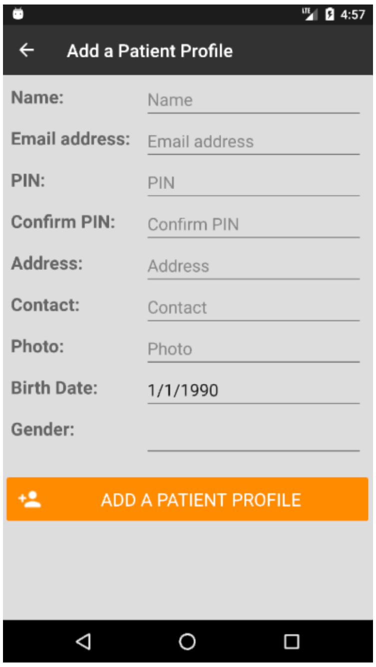

Figure 6.20 - Add Physiotherapist Profile page.

In the Home Page it is also possible to select the profile that we want to see and open it. In the View Profiles page (see Figure 6.21) we can see all the information regarding the physiotherapists or patients and edit or delete the profiles. The Edit Profiles page is identical to the Edit My Profile page (see figure 6.16). When the logged user is a physiotherapist, and opens a patient profile, is allowed on this page. Besides the possibility of changing the patient information and including the patient in his favourite list, physiotherapists can also visualize information about the patient's exercise plans (see figure 6.22).

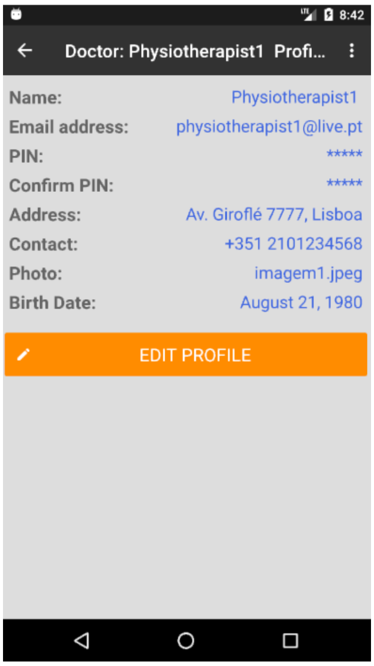

Figure 6.21 - View Physiotherapist Profile page.

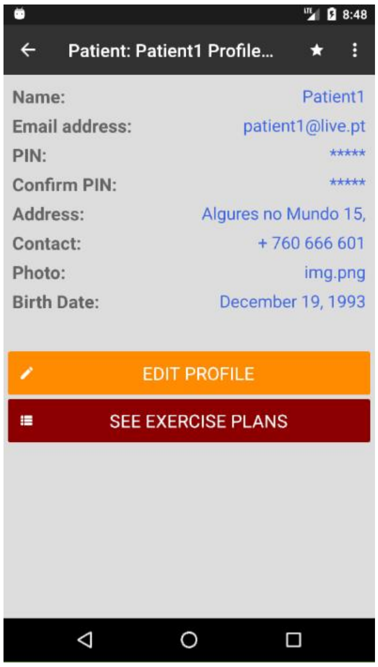

Figure 6.22 - View Patient Profile page.

If the physiotherapist opens the Exercise Plans page, a list of the exercise plans of the patient is presented, if there are any, like it is shown in Figure 6.23. The patient can open this page from their Home Page. It is possible for the Physiotherapist to create a new Exercise Plan (see Figure 6.24) or open an existing one (see Figure 6.25).

Figure 6.23 - View Exercise Plans page.

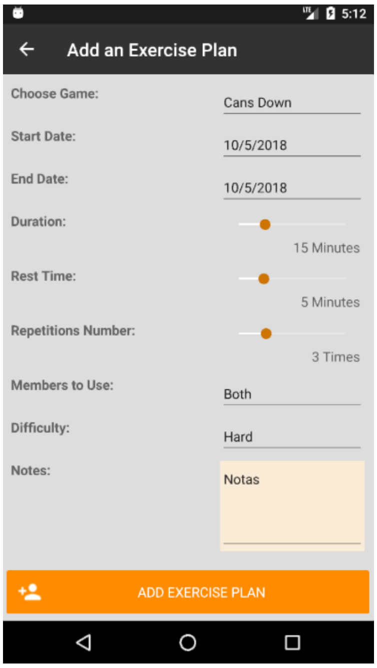

Figure 6.24 - Add Exercise Plan page.

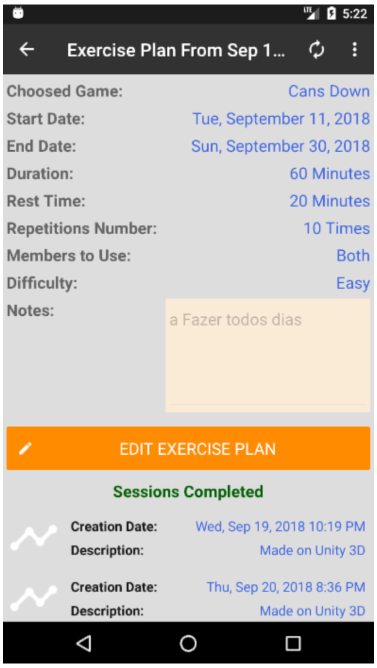

Figure 6.25 - Exercise Plan page.

The Exercise Plan page allows the physiotherapist to edit the selected exercise plan (see Figure 6.26) and visualize a list of all sessions created for the patient in the game platform associated with the exercise plan selected. Patients have access to the same page to view their own data sessions, on Home Page. In the Session Report page, physiotherapists and patients, can see all results, such as collected data of the members used and gaming scores (see Figure 6.27). In this page it is also possible to see the charts data associated with members and head movements on gaming performance, by clicking the View Charts of Session button. In the charts it is possible to select the data to show, by clicking the correspondent button (see Figure 6.28).

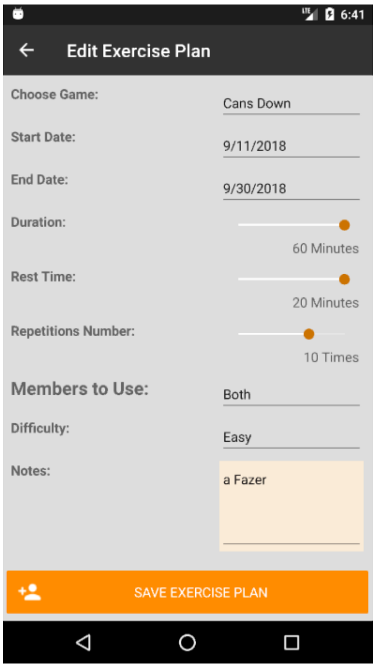

Figure 6.26 - Edit Exercise Plan page.

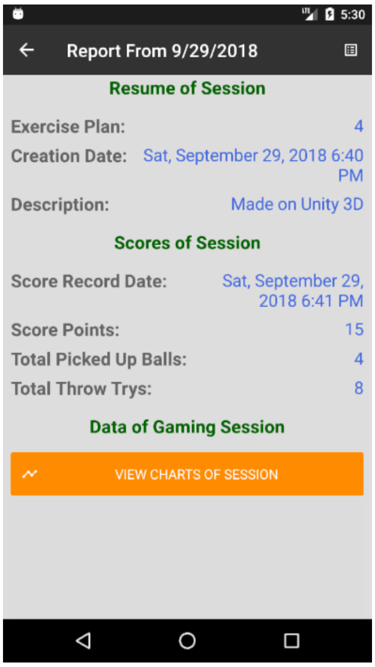

Figure 6.27 - Session Report page.

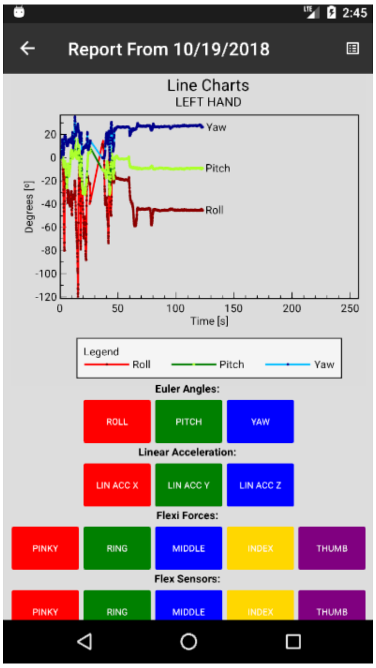

Figure 6.28 - Session Report page showing charts.

In the right upper corner of the Session Report page exists a button to access the Observation page. The Observation page is for the Physiotherapist to write some notes or observations around the exercise session performed (see Figure 6.29), making it visible to the patient. This information can be changed whenever the Physiotherapist decides. The Patient can see this information at the same location in the application (see Figure 6.30).

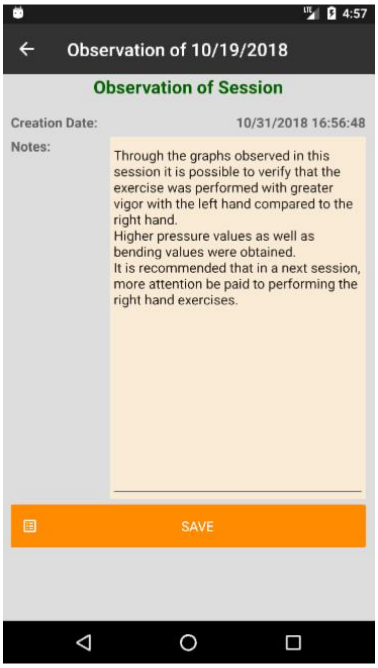

Figure 6.29 - Observation page created by Physiotherapist.

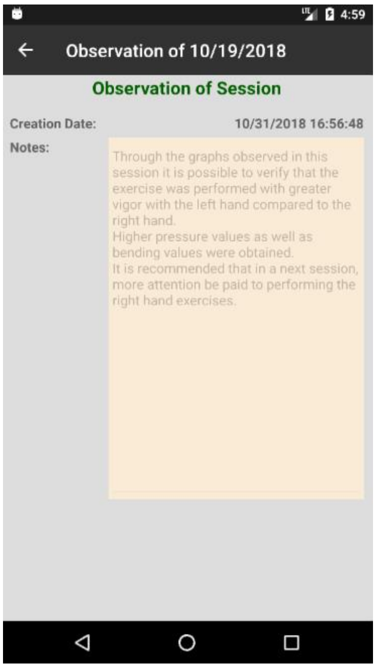

Figure 6.30 - Observation page accessed by Patient.

This covers all the main features of the mobile application. All images were captured in the mobile application Physio Wear created in Xamarin.Forms and running at a mobile phone with Android OS.

The application was tested in the version for Windows, being equal in the sequence of actions. The main change is that this features the Windows 10 Native controls. The iOS version was not tested due to lack of equipment.

Chapter 7 - Results

Overview

This chapter demonstrates the experimental results associated with Physio Wear system tests.

The tests were performed in a first phase with healthy volunteers. The performed tests were conducted to get results and try to understand people's feedback when playing games in the Unity application or using the mobile application and to have information that can improve the serious game quality for future adoption by the physiotherapy clinics and their patients.

7.1 Preliminary Results

In an intermediate phase of the development of the Unity application with the Cans Down Game some test sessions were conducted with healthy volunteers and the obtained results were reported in a publication indexed IEEEXplore [see *Appendix A - Article*] and presented on *International Symposium on Sensing and Instrumentation in IoT Era 2018*, in China, by Professor Octavian Postolache. The article was awarded at this conference with Topic Student Best Paper and presented in Appendix A. This publication aimed to report the development of the project, motivational factors for those using the system as well as the primary analysis of data collected from the sensor devices generated in the Sensor Layer, captured by Edger Layer (Arduino Platform), processed in Gaming Layer and later recorded in the Cloud Database Layer.

At the time of the article's creation, only the right-hand prototype smart-glove had been developed. The preliminary tests related, with the serious game implemented, were performed in an ISCTE-IUL classroom with two young male healthy volunteers (25 years and 26 years) and height (1.78 m and 1.75 m), respectively. The results obtained are presented in the following charts, Figure 7.1 and Figure 7.2:

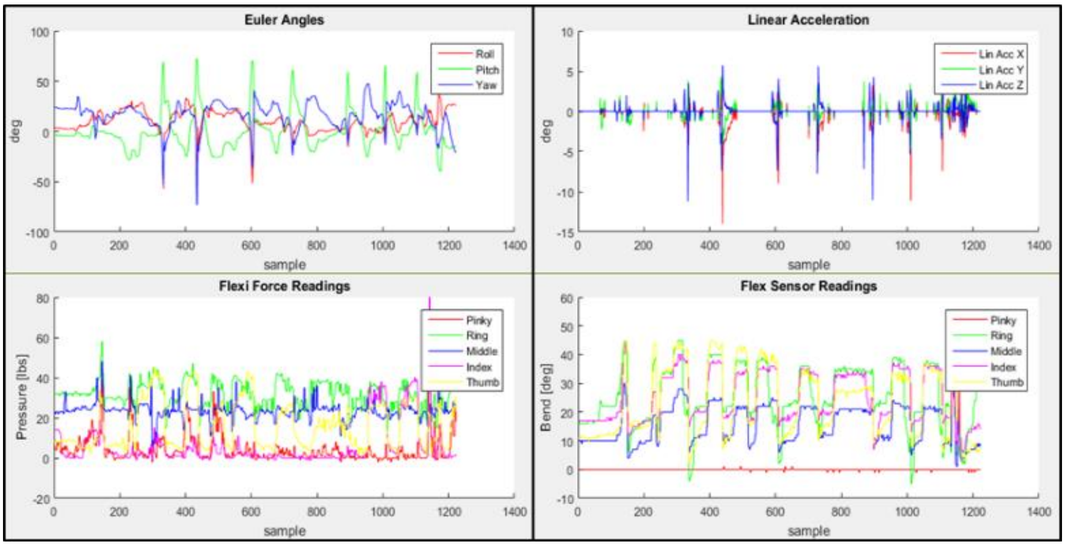

Figure 7.1 - Example of data extracted from tests of participant 1 with Right Hand Glove on Cans Down using MatLAB.

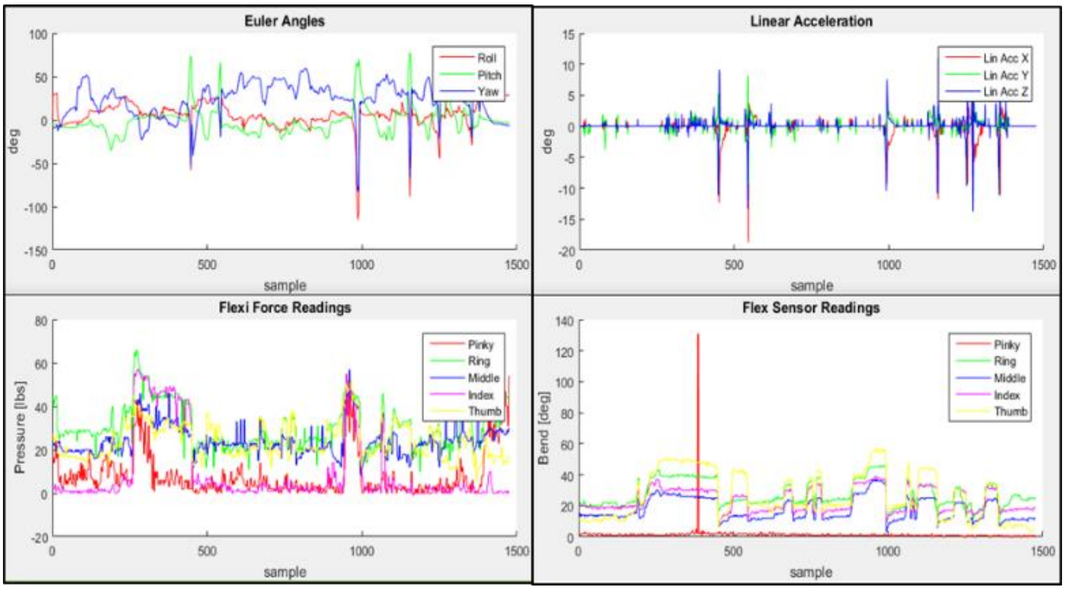

Figure 7.2 - Example of data extracted from tests of participant 2 with Right Hand Glove on Cans Down using MatLAB.

The tests performed by participant 1 lasted for 85 seconds. Through the charts presented in Figure 7.1, it is possible to observe, through the Euler angles variations, that 7 attempts were made to strike the virtual balls, as there were peaks in the Pitch value. It is also possible to observe all the variations of Linear Acceleration in the 3 Cartesian components, where there is a greater variation in the Z-axis. The Flexi Force readings allows us to observe all the forces applied at the tips of each finger, in the movements of opening and closing the hand, where it can be verified that Index Finger was the one that captured higher values during the exercise. This value is estimated for a range of 0-100 lbs. The Flex Sensor readings chart allows us to visualize all the angular flexures performed during the exercise, the values being estimated for a range of 0-90 degrees.

The tests performed by participant 2 had a duration of 103 seconds and are presented in Figure 7.2. The charts present the Euler angles variations for 6 attempts that were made to strike the virtual balls, highlighted by the evolution of Pitch value.

In both tests it was verified that the reading value collected from the FlexSensor, referring to the pinkie finger is practically null because the sensor has been damaged.

7.2 Data Analysis

Several testing sessions were carried out in laboratory with a few two healthy volunteers, both males with 25 years old, to evaluate the performance of the serious games for virtual reality and real scenarios. The healthy volunteers have played the both serious games developed for both game scenarios in different sessions of exercises. At the end of the usability tests, hand and finger data were collected by the application for further analysis. The following graphs presents some of the data collected by the Unity application and processed in Microsoft Excel. The data collected are always described in relation to the time, the considered tests period being expressed by 3 minutes = 180 seconds.

Between the tests performed by participant 1 and participant 2 a recalibration was performed on magnetometer present in the IMU of the devices. This translates into some discrepancy in the fineness of the Euler Angles and Linear Acceleration readings.

The participant 1 has grabbed a total of 19 balls during the exercise of Cans Down Game and 8 cans was knocked down. The participant 2 has grabbed a total of 7 balls with Right hand and can touch in 8 ping-pong paddles with Left hand. A few 7 cups were knocked down during the exercise of Coffee Pong Game.

Figure 7.3 represents the charts for the both serious games played with both hands for two participants with values referring to Euler Angles over time (seconds).

Participant 1 - Cans Down Game **Participant 2 - Coffee Pong Game**

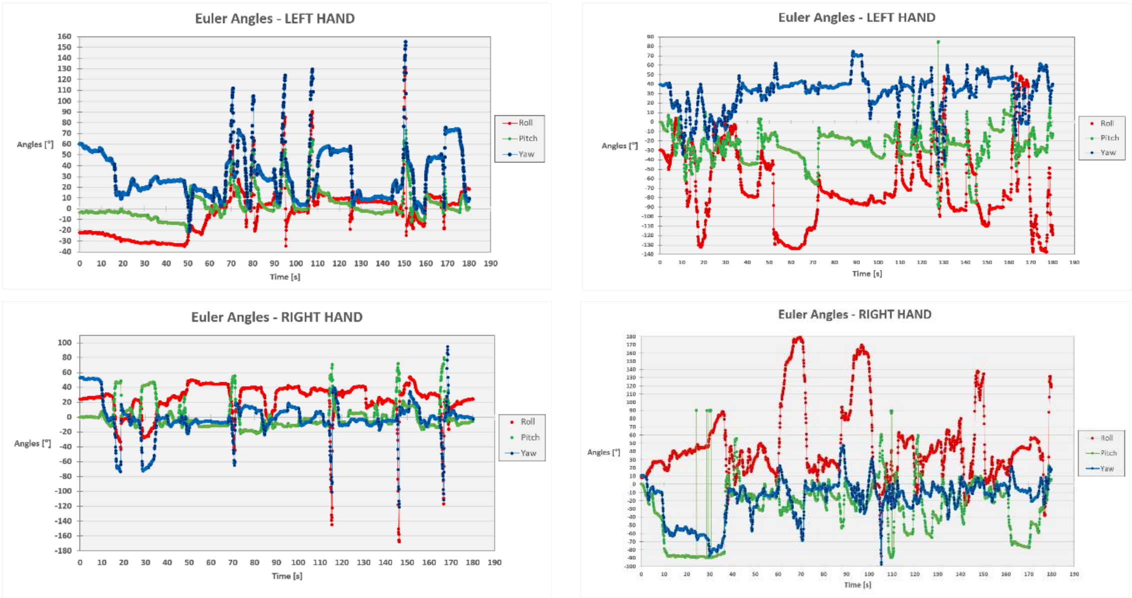

Figure 7.3 - Left and Right hand charts of two participants, with values of Euler Angles for both Serious Games.

- Euler Angles – Indicates values measured in degrees between -180º and 180º of the rotation of the wrist;

Through the values of rotation of the members it is possible to verify that participant 1 made 5 attempts to throw the balls with the left hand and 6 with the right hand. The number of throwing attempts is visible in the peaks recorded in the graphs, because in each throw attempt there is a big change in the Pitch and Yaw values.

The participant 2 presented a greater variation in the rotation of the limbs, which shows that this member had more difficulty grasping the objects. The roll spikes recorded on the right hand show that the patient rotated the wrist more than once in an attempt to catch the ball. The pitch peaks recorded for the 40, 110 and 120 seconds evidence attempts to throw the ball with the right hand. The left hand rescued 1 peak in the pitch value at 125 seconds, which resulted in an unsuccessful attempt to grab a paddle.

Figure 7.4 represents the charts for the both serious games played with both hands for two participants with values referring to Linear Acceleration over time (seconds).

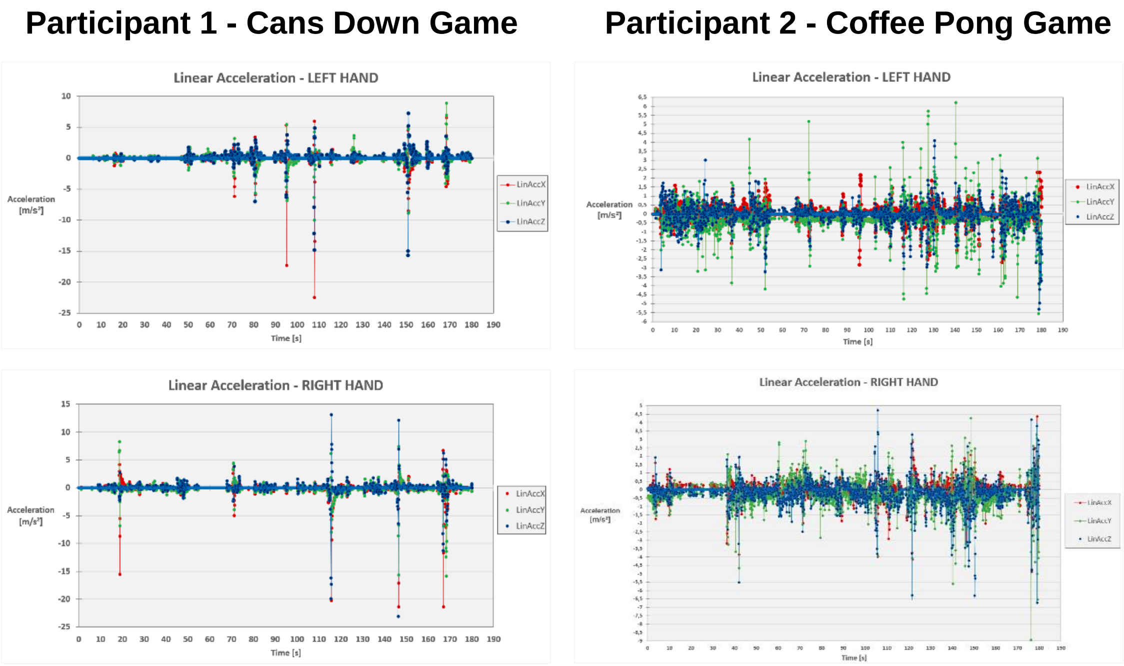

Figure 7.4 - Left and Right hand charts of two participants, with values of Linear Acceleration for both Serious Games.

- Linear Acceleration – Indicates values measured in m/s² of the linear acceleration movements of wrists;

The linear acceleration values of the members show the above that participant 1 made 5 attempts to throw the balls with the left hand and 6 with the right hand. The number of throwing attempts is visible in the peaks recorded in the graphs, because in each throw attempt there is a great change in the acceleration values in the 3 Cartesian components, XYZ.

Participant 2 presented a greater variation in the rotation of the limbs, which shows that the limb had more difficulty in grasping the objects. The values recorded at the level of the linear acceleration variation, show a greater difficulty for the participant to meet the objectives. The graphs of both hands show a greater dispersion of values.

Figure 7.5 represents the charts for the both serious games played with both hands for two participants with values referring to FlexiForce readings over time (seconds).

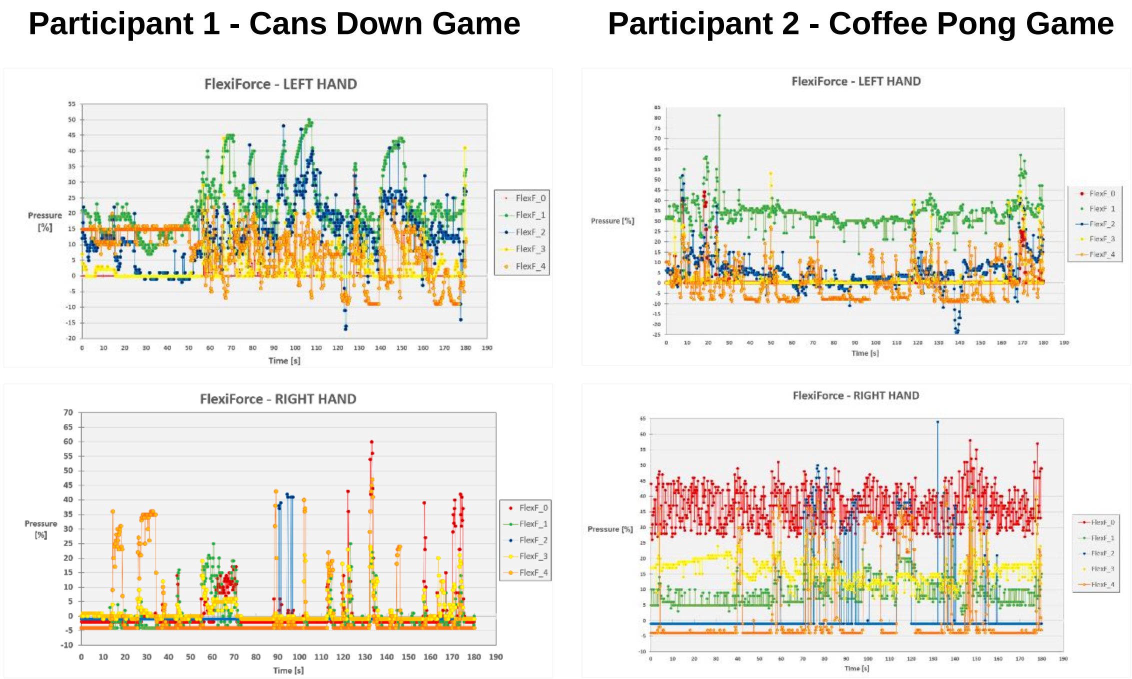

Figure 7.5 - Left and Right hand charts of two participants, with values of FlexiForce Readings for both Serious Games.

readings range of the sensor;

- The FlexiF_0 readings correspond to Pinky Finger;
- The FlexiF_1 readings correspond to Ring Finger;
- The FlexiF_2 readings correspond to Middle Finger;
- The FlexiF_3 readings correspond to Index Finger;
- The FlexiF_4 readings correspond to Thumb Finger;

At the level of the reading of the piezo resistive sensors, in relation to FlexiForce, the values recorded show that the participant 1 exerted more pressure points, referring to the left hand, over the thumb, middle and ring fingers, as they were the points that registered higher values, and those that most occur. The right hand, on the other hand, recorded more pressure variations on the pinky and thumb fingers, in the moments of grabbing the virtual objects. Participant 2 presented a greater variation of the pressure points on the thumb and ring fingers, for the left hand, and for the right hand on the pinky, middle and thumb fingers.

Figure 7.6 represents the charts for the both serious games played with both hands for two participants with values referring to FlexSensor readings over time (seconds).

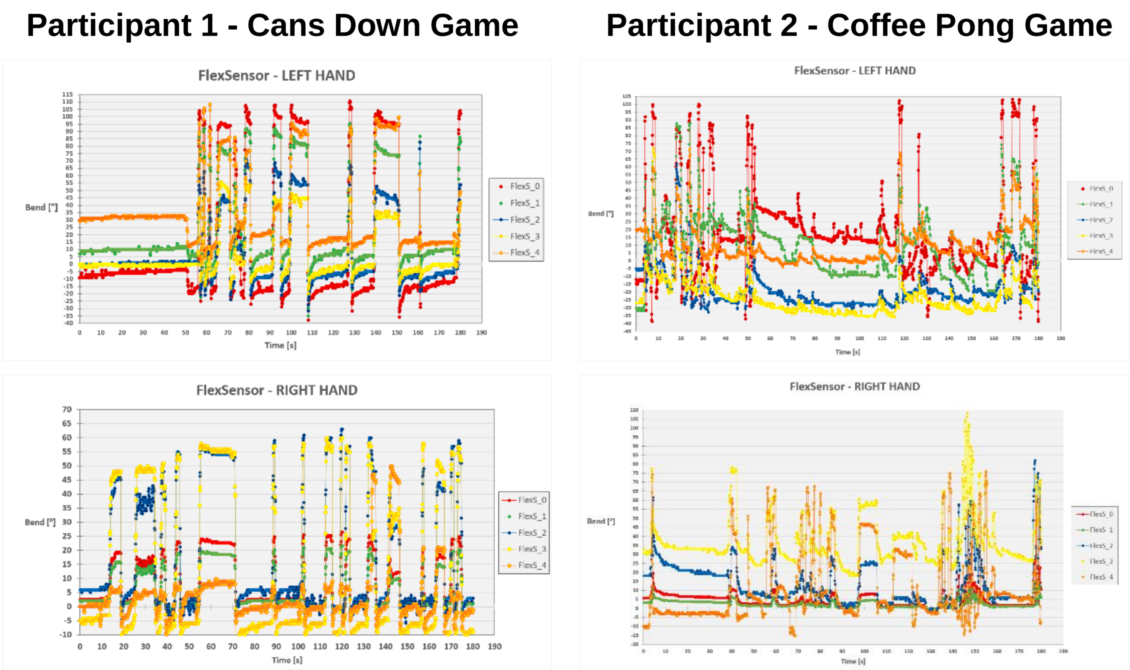

Figure 7.6 - Left and Right hand charts of two participants, with values of FlexSensor Readings for both Serious Games.

- FlexSensor Readings – Indicates values measured in degrees between -5º and 90º of flexion of readings by sensor;
- The FlexS_0 readings correspond to Pinky Finger;
- The FlexS_1 readings correspond to Ring Finger;
- The FlexS_2 readings correspond to Middle Finger;
- The FlexS_3 readings correspond to Index Finger;
- The FlexS_4 readings correspond to Thumb Finger;

Regarding the readings of the FlexSensor piezo resistive sensors, the values obtained show that the participant 1 exerted flexion points on all the fingers, with the values recorded on the pinky and thumb fingers referring to the left hand, those that registered higher values and greater variation. In the right hand, on the other hand, there was a greater variation of the flexion points on the middle and index fingers, in the moments of grasping the virtual objects. Participant 2 presented a greater variation of flexion points on the pinky, index and thumb fingers for the left hand, and for the right hand on the index finger and thumb.

Figure 7.7 represents the charts of the head for the both serious games for two participants with values referring to Euler Angles and Linear Accelerations over time (seconds). In addition to being used in the process of navigation in the game scene, these data can be used to evaluate the user's posture during the exercises.

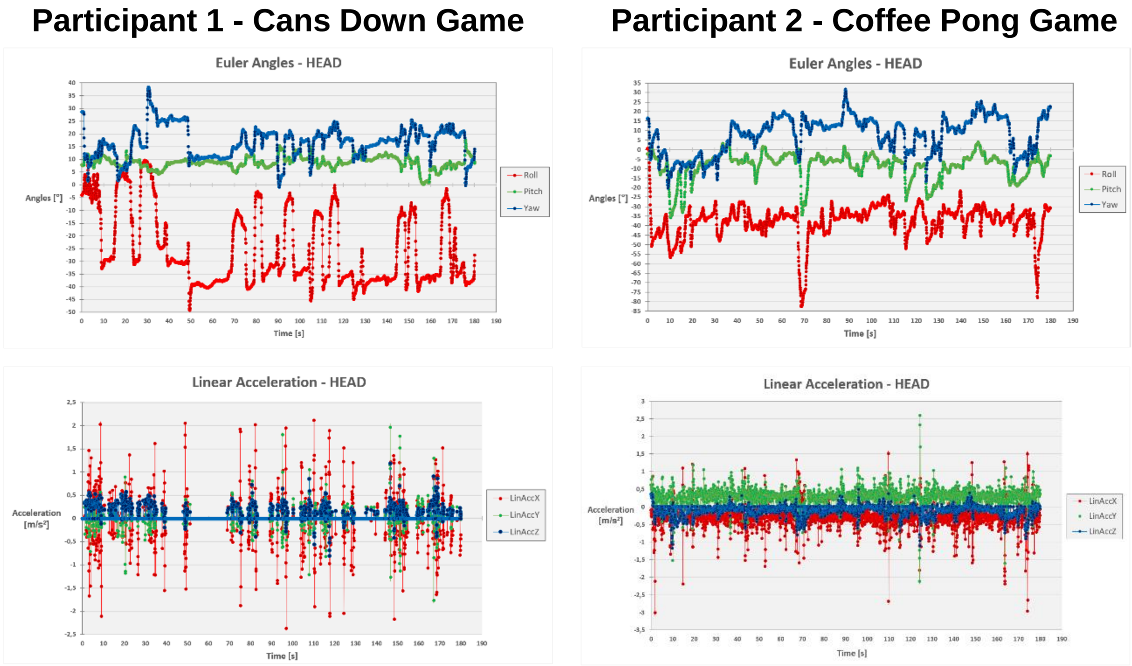

Figure 7.7 - Head charts of two participants, with values of Euler Angles and Linear Accelerations for both Serious Games.

- Euler Angles – Indicates values measured in degrees between -180º and 180º of the rotation of the head;
- Linear Acceleration – Indicates values measured in m/s² of the linear acceleration movements of the head;

Regarding the readings of the devices that collect data from the head, the values acquired show that the variation of the rotation values was very similar for both participants, however, there was a difference in the values of linear acceleration of the graph of participant 1 to that of the participant 2. This difference is explained by the fact that the participant 2 has had more difficulty establishing the reference point (calibration) with the game monitor, leading to a greater dispersion of linear acceleration readings.

The previous charts have been processed in Microsoft Excel, but also can be visualized on the mobile application. Figure 7.8 illustrates a comparison between the data processed in Excel and by mobile application.

Excel **Xamarin Android App**

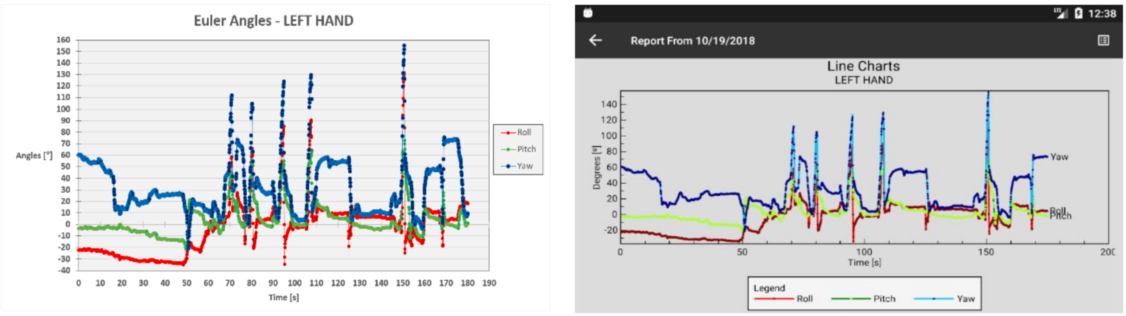

Figure 7.8 - Comparison between the data processed in Excel and by mobile application.

Chapter 8 - Conclusions and Future Work

Overview

This chapter presents the conclusions and future work of the project that was presented in the previous chapters.

8.1 Conclusions

This dissertation consists of a system that was developed to be used in physiotherapy, specifically in physical and motor rehabilitation of hands and fingers. The designed technology aims to help healthcare professionals improve their work by effectively monitoring patients and objectively evaluating the results of physiotherapy sessions.

The system consists of a serious therapeutic gaming system that combines wearable sensors and VR scenarios. The used technology for the VR component was Unity 3D with different scenarios being carried out. The developed system, is characterized by an IoT architecture including force and IMU sensors deployed on a glove pair and a headband and connected to a microcontroller as edge computation platform.

The development phase of the project began with a survey to better understand how serious games should be created and adapted to the needs and preferences of each patient. Subsequently, serious games were developed to respond to those needs and preferences. Serious games have been developed through the Unity 3D gaming engine and integrated with developed wearable devices, smart gloves and smart headband.

The system provides two main tools, a game application that allows patients to perform therapy exercises by playing serious games in virtual reality environments, and another for physiotherapists to visualize the results and manage patients. Patients will also be able to access your data in the mobile application. All the most relevant information, such as exercise scores or movement data collected, is stored on a remote server, which allows communication between the two applications.

Using this system, the physiotherapist can follow the patient during the prescribed training plan for days, weeks or months. Considering the recorded data associated with the patients the data analytics can be applied to extract the rehabilitation model and to predict the rehabilitation outcome.

On the other hand, this system also seeks to constitute a high source of motivation for the main users and beneficiaries of the system, patients, who for some reason need to carry out rehabilitation programs, through the high interactivity provided by the inclusion of movements of the patient in the VR scenario by the created wearables.

The mobility characteristics of the system allows the patient to stay in his home and to perform the exercises proposed by the physiotherapist. Easy installation of the system and calibration of the device by non-specialists for use in a non-clinical environment, make this solution suitable for use at home. Patients feel psychologically better in their own environment and also the speed of rehabilitation is improved.

Constrains

After completing the usability tests of the system and the Game platform, some constraints have been highlighted. Both participants concluded that the quality of the piezo resistive sensors prevented a better data acquisition, also causing a noticeable impact on the gameplay, because the movements related to the process of opening the hands in the virtual scenario are totally dependent on the analog readings recorded by the FlexSensor. The process of putting on and removing the gloves has proved to be detrimental enough to maintain the electrical connections between the wearable devices and the said Flex Sensors, damaging the connections themselves several times.

In the future, a better connection will be considered in order to make the readings acquired more reliable and make the gameplay even more interesting from the user's point of view.

8.2 Future Work

The project was long and challenger keeping many parts that can be improved in the future. This way some points can be mentioned:

- Performing of an extended study on regarding the selection of the movements that are more appropriate for finger and hand rehabilitation. Through the study, will be performed new games with different themes, allowing a greater variety of exercising plans;

- Make serious games even more adaptable to patient characteristics increasing the game personalization, in addition, an improvement of the own virtual environments would make the exercises even more interactive for those who use the system;
- During the realization of the user's manual of the system it was concluded that the magnetometer calibration process (necessary for a good performance in the acquisition of the IMU readings) proves to be very demanding at the technical level for the ordinary user because it is a prototype. It will be considered in the future to carry out an optimization of this process, making it simpler for the user.
- Include a greater detail in the game's scoring system, providing a better evaluation standard on each member used in a given session;
- Include a more detailed statistical and evolutionary analysis around the sessions performed (including analysis of maximum and minimum values, variances, dispersions and averages of all data acquired by the system);
- Perform an improved, extensive and adapted evolutive Report according to each patient and the needs of physiotherapists;
- Improvements to the mobile application user's viewpoint, especially on the session result pages;
- The possibility to apply Big Data Analysis for the data obtained with developed system and to develop a set of personalized rehabilitation models;
- The creation of new games for lower limb rehabilitation will be considered in the future, as well the inclusion of new sensors in system.

Despite all the improvements that can be made to the project, the solution already implemented represents a valid and ready solution to be used in physiotherapy clinics or in patients' homes to assist people in the motor rehabilitation of the hands and fingers, but also for the physiotherapists to have a tool to analyse the results of rehabilitation sessions in objective way.

References

[1] Learn.org. (2018). *What Is Health Science?*. [online] Available at: https://learn.org/articles/What_is_Health_Science.html [Accessed 8 Mar. 2018].

[2] Pt.wikipedia.org. (2018). *Ciências da saúde*. [online] Available at: https://pt.wikipedia.org/wiki/Ciências_da_saúde [Accessed 8 Mar. 2018].

[3] Unric.org. (2018). *Envelhecimento*. [online] Available at: https://www.unric.org/pt/envelhecimento [Accessed 2 Mar. 2018].

[4] MedXForce. (2018). [online] Available at: https://medxforce.com/stroke/Statistics/ [Accessed 2 Mar. 2018].

[5] McKinsey & Company. (2018). *Unlocking the potential of the Internet of Things*. [online] Available at: https://www.mckinsey.com/business-functions/digital-mckinsey/our-insights/the-internet-of-things-the-value-of-digitizing-the-physical-world [Accessed 2 Mar. 2018].

[6] O. Postolache and P. S. Girão, "Physiotherapy smart connected devices for S-health," 2016 7th International Conference on Information, Intelligence, Systems & Applications (IISA), Chalkidiki, 2016, pp. 1-6.

[7] C. Watters, S. Oore, M. Shepherd, A. Abouzied, A. Cox, M. Kellar, H. Kharrazi, F. Liu, A. Otley, "Extending the Use of Games in Health Care", Proc. 39th Hawaii International Conference on System Sciences (HICSS), January 2006.

[8] N. Suttiea, S. Louchart, T. Lim, A. Macvean, W. Westera, D. Brown, D. Djaouti, "Introducing the "Serious Games Mechanics. A Theoretical Framework to Analyse Relationships Between Game and Pedagogical Aspects of Serious Games", Procedia Computer Science, Vol 15, pp 314–315, 2012.

[9] Chang, C-Y.; Lange, B.; Zhang, M.; Koenig, S.; Requejo, N.; Sawchuk, A.A.; Rizzo, A.A.; Towards Pervasive Physical Rehabilitation using Microsoft Kinect, Proc. 6th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth), San Diego – CA, USA, May 2012.

[10] T. Susi, M. Johannesson, and P. Backlund, "Serious games—An overview," School of Humanities and Informatics, University of Skovde, ¨ Skovde, Sweden, 2007.

[11] Djaouti, Damien; Alvarez, Julian; Jessel, Jean-Pierre. "Classifying Serious Games: the G/P/S model". Ludoscience.com. (2018). [online] Available at: http://www.ludoscience.com/files/ressources/classifying_serious_games.pdf [Accessed 12 May 2018].

[12] Pt.wikipedia.org. (2018). *Food Force*. [online] Available at: https://pt.wikipedia.org/wiki/Food_Force [Accessed 15 May 2018].

[13] BreakAway Games. (2018). *vHealthCare - BreakAway Games*. [online] Available at: http://www.breakawaygames.com/vhealthcare/ [Accessed 15 May 2018].

[14] Mursion.com. (2018). *Mursion - Virtual Reality Simulations for Learning*. [online] Available at: https://mursion.com/ [Accessed 17 Apr. 2018].

[15] Mirarehab.com. (2018). MIRA - Play your way to recovery. [online] Available at: http://www.mirarehab.com/ [Accessed 23 Jun. 2018].

[16] K. A. Ericsson, “The influence of experience and deliberate practice on the development of superior expert performance,” in The Cambridge Handbook of Expertise and Expert Performance. K. A. Ericsson, N. Charness, P. J. Feltovich, and R. R. Hoffman, Eds. Cambridge, U.K.: Cambridge, Univ. Press, 2006, pp. 685–705.

[17] En.wikipedia.org. (2018). *Exergaming*. [online] Available at: https://en.wikipedia.org/wiki/Exergaming [Accessed 18 Apr. 2018].

[18] T. Mettler and R. Pinto, "Serious Games as a Means for Scientific Knowledge Transfer—A Case From Engineering Management Education," in IEEE Transactions on Engineering Management, vol. 62, no. 2, pp. 256-265, May 2015.

[19] J. Wiemeyer and A. Kliem, "Serious games in prevention and rehabilitation - a new panacea for elderly people?," European Group for Research into Elderly and Physical Activity, vol. 9, pp. 41-50, 2012.

[20] Serge Timsit, Stroke at the beginning of the XXIst century, La Presse Médicale, Volume 45, Issue 12, Part 2, 2016, Pages e389-e390, ISSN 0755-4982

[21] Yannick Béjot, Henri Bailly, Jérôme Durier, Maurice Giroud, Epidemiology of stroke in Europe and trends for the 21st century, La Presse Médicale, Volume 45, Issue 12, Part 2.2016, Pages e391-e398, ISSN 0755- 4982

[22] Ma M., Zheng H. (2011) Virtual Reality and Serious Games in Healthcare. In: Brahnam S., Jain L.C. (eds) Advanced Computational Intelligence Paradigms in Healthcare 6. Virtual Reality in Psychotherapy, Rehabilitation, and Assessment. Studies in Computational Intelligence, vol 337. Springer, Berlin, Heidelberg

[23] Nvidia.com. (2018). NVIDIA PhysX System Software. [online] Available at: https://www.nvidia.com/object/physx-9.17.0524-driver.html [Accessed 2 May 2018].

[24] Unrealengine.com. (2018). What is Unreal Engine 4. [online] Available at: https://www.unrealengine.com/en-US/what-is-unreal-engine-4 [Accessed 2 May 2018].

[25] Unity. (2018). Unity. [online] Available at: https://unity3d.com/pt [Accessed 2 May 2018].

[26] Nintendo of Europe GmbH. (2018). *Wii Sports*. [online] Available at: https://www.nintendo.pt/Jogos/Wii/Wii-Sports-283971.html [Accessed 29 May 2018].

[27] Dora Celinder & Hanne Peoples (2012) Stroke patients' experiences with Wii Sports® during inpatient rehabilitation, Scandinavian Journal of Occupational Therapy, 19:5, 457-463

[28] Kerrigan, D.J., Chen, H., Wiederhold, M.D., Gamberini, L., Wiederhold, B.K.: Evaluation of the Nintendo Wii for Physical Exercise and Rehabilitation. Cyber Therapy 13 (June 23- 25, 2008)

[29] M. Rodgers, "Wearable Sensors & Systems with Application in Rehabilitation," [Online] Available at: http://www.medicalautomation.org/wp-content/uploads/2012/11/Rodgers_Sensors_hopkins.pdf [Accessed July 2018].

[30] En.wikipedia.org. (2018). *PlayStation Move*. [online] Available at: https://en.wikipedia.org/wiki/PlayStation_Move [Accessed 1 Jun. 2018].
[31] En.wikipedia.org. (2018). *Kinect*. [online] Available at: https://en.wikipedia.org/wiki/Kinect [Accessed 1 Jun. 2018].
[32] Leap Motion. (2018). *Home - Leap Motion*. [online] Available at: https://www.leapmotion.com/ [Accessed 2 Jun. 2018].
[33] Playstation. (2018). *PlayStation®VR*. [online] Available at: https://www.playstation.com/pt-pt/explore/playstation-vr/ [Accessed 2 Jun. 2018].
[34] Vive.com. (2018). *VIVE™ | Discover Virtual Reality Beyond Imagination*. [online] Available at: https://www.vive.com/us/ [Accessed 1 Apr. 2018].
[35] O. Postolache, F. Cary, P. S. Girão and N. Duarte, "Physiotherapy assessment based on Kinect and mobile APPs," 2015 6th International Conference on Information, Intelligence, Systems and Applications (IISA), Corfu, 2015, pp. 1-6.
[36] F. Lourenço, O. Postolache and G. Postolache, "Tailored virtual reality and mobile application for motor rehabilitation," 2018 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), Houston, TX, 2018, pp. 1-6.
[37] Alavi, Shamir, Dennis Arsenault, and Anthony Whitehead. “Quaternion-Based Gesture Recognition Using Wireless Wearable Motion Capture Sensors.” Ed. Fabrizio Lamberti, Andrea Sanna, and Jon Rokne. Sensors (Basel, Switzerland) 16.5 (2016): 605. PMC. Web. 20 Mar. 2018
[38] Kim A., Golnaraghi M. Initial calibration of an inertial measurement unit using an optical position tracking system; Proceedings of the Position Location and Navigation Symposium (PLANS 2004); Monterey, CA, USA. 26–29 April 2004.
[39] Zhou, H., Hu, H.: A survey - human movement tracking and a research proposal, Technical Report (2004)
[40] Historyofinformation.com. (2018). *The Sayre Glove (1977) : HistoryofInformation.com*. [online] Available at: https://historyofinformation.com/expanded.php?id=4080 [Accessed 1 May 2018].
[41] En.wikipedia.org. (2018). *VPL Research*. [online] Available at: https://en.wikipedia.org/wiki/VPL_Research [Accessed 17 May 2018].
[42] En.wikipedia.org. (2018). *Power Glove*. [online] Available at: https://en.wikipedia.org/wiki/Power_Glove [Accessed 17 May 2018].
[43] CyberGlove Systems LLC. (2018). *Home*. [online] Available at: http://www.cyberglovesystems.com/ [Accessed 16 May 2018].
[44] 5dt.com. (2018). *Data Gloves | 5DT*. [online] Available at: http://www.5dt.com/data-gloves/ [Accessed 16 May 2018].
[45] Qualisys.com. (2018). Qualisys | Motion Capture Systems. [online] Available at: http://www.qualisys.com/ [Accessed 12 Jul. 2018].
[46] VICON. (2018). Motion Capture Systems. [online] Available at: http://www.vicon.com/ [Accessed 12 Jul. 2018].
[47] Codamotion. (2018). Home - Codamotion. [online] Available at: http://www.codamotion.com/ [Accessed 12 Jul. 2018].

[48] Xsens 3D motion tracking. (2018). Home - Xsens 3D motion tracking. [online] Available at: https://www.xsens.com/ [Accessed 12 Jul. 2018].
[49] Sqlite.org. (2018). SQLite Home Page. [online] Available at: http://www.sqlite.org/ [Accessed 2 May 2018].
[50] Mysql.com. (2018). MySQL. [online] Available at: https://www.mysql.com/ [Accessed 2 May 2018].
[51] S. Hiremath, G. Yang and K. Mankodiya, "Wearable Internet of Things: Concept, architectural components and promises for person-centered healthcare," 2014 4th International Conference on Wireless Mobile Communication and Healthcare - Transforming Healthcare Through Innovations in Mobile and Wireless Technologies (MOBIHEALTH), Athens, 2014, pp. 304-307.)
[52] Play.google.com. (2018). Google Fit. [online] Available at: https://play.google.com/store/apps/details?id=com.google.android.apps.fitness&hl=pt_PT [Accessed 2 Feb. 2018].
[53] Play.google.com. (2018). Nike+ Run Club. [online] Available at: https://play.google.com/store/apps/details?id=com.nike.plusgps [Accessed 2 Feb. 2018].
[54] Ludwin, S., & Greysen, S. R. (2015). "Use of Smartphones and Mobile Devices in Hospitalized Patients: Untapped opportunities for inpatient engagement". Journal of Hospital Medicine, 10(7), 459–461.
[55] Kaa IoT platform. (2018). Kaa Enterprise IoT platform. [online] Available at: https://www.kaaproject.org/ [Accessed 2 Aug. 2018].
[56] Learn.sparkfun.com. (2018). Voltage Dividers - learn.sparkfun.com. [online] Available at: https://learn.sparkfun.com/tutorials/voltage-dividers [Accessed 2 Aug. 2018].
[57] Learn.sparkfun.com. (2018). Analog to Digital Conversion - learn.sparkfun.com. [online] Available at: https://learn.sparkfun.com/tutorials/analog-to-digital-conversion [Accessed 2 Aug. 2018].
[58] MPU-9250, S. (2018). SparkFun IMU Breakout - MPU-9250 - SEN-13762 - SparkFun Electronics. [online] Sparkfun.com. Available at: https://www.sparkfun.com/products/13762 [Accessed 2 Aug. 2018].
[59] Tekscan. (2018). FlexiForce A201 Sensor. [online] Available at: https://www.tekscan.com/products-solutions/force-sensors/a201 [Accessed 18 May 2018].
[60] Spectrasymbol.com. (2018). Flex Sensors (Resistive) – Spectra Symbol – Innovation-Driven. [online] Available at: http://www.spectrasymbol.com/product/flex-sensors/ [Accessed 18 May 2018].
[61] Electrofun. (2018). Módulo Bluetooth HC-05 para Arduino. [online] Available at: https://www.electrofun.pt/comunicacao/modulo-bluetooth-arduino-hc05 [Accessed 19 May 2018].
[62] Store.arduino.cc. (2018). Arduino Nano. [online] Available at: https://store.arduino.cc/arduino-nano [Accessed 13 May 2018].

[63] Kim A., Golnaraghi M. Initial calibration of an inertial measurement unit using an optical position tracking system; Proceedings of the Position Location and Navigation Symposium (PLANS 2004); Monterey, CA, USA. 26–29 April 2004.
[64] En.wikipedia.org. (2018). Attitude and heading reference system. [online] Available at: https://en.wikipedia.org/wiki/Attitude_and_heading_reference_system [Accessed 2 May 2018].
[65] Zhi, Ruoyu, "A Drift Eliminated Attitude & Position Estimation Algorithm In 3D" (2016). Graduate College Dissertations and Theses. 450. [Online] https://scholarworks.uvm.edu/graddis/450 [Accessed 15 May, 2018]
[66] Mathworld.wolfram.com. (2018). Euler Angles -- from Wolfram MathWorld. [online] Available at: http://mathworld.wolfram.com/EulerAngles.html [Accessed 5 Jun. 2018].
[67] Upload.wikimedia.org. (2018). [online] Available at: https://upload.wikimedia.org/wikipedia/commons/c/c1/Yaw_Axis_Corrected.svg [Accessed 5 Jun. 2018].
[68] GitHub. (2018). richardstechnotes/RTIMULib-Arduino. [online] Available at: https://github.com/richardstechnotes/RTIMULib-Arduino [Accessed 2 Apr. 2018].
[69] GitHub. (2018). kriswiner/MPU9250. [online] Available at: https://github.com/kriswiner/MPU9250 [Accessed 2 Apr. 2018].
[70] Nxp.com. (2018). [online] Available at: https://www.nxp.com/files-static/sensors/doc/app_note/AN3461.pdf [Accessed 14 May 2018].
[71] Blog.tkjelectronics.dk. (2018). TKJ Electronics » A practical approach to Kalman filter and how to implement it. [online] Available at: http://blog.tkjelectronics.dk/2012/09/a-practical-approach-to-kalman-filter-and-how-to-implement-it/ [Accessed 14 Apr. 2018].
[72] Assetstore.unity.com. (2018). Unity Asset Store - The Best Assets for Game Making. [online] Available at: https://assetstore.unity.com/ [Accessed 2 May 2018].
[73] Uniduino.com. (2018). Getting Started » Uniduino. [online] Available at: http://www.uniduino.com/kb/getting-started/ [Accessed 1 Aug. 2018].
[74] Zucconi, A. and Zucconi, A. (2018). How to integrate Arduino with Unity - Alan Zucconi. [online] Alan Zucconi. Available at: https://www.alanzucconi.com/2015/10/07/how-to-integrate-arduino-with-unity/ [Accessed 1 Aug. 2018].
[75] L. Motion, "LeapMotionCoreAssets," GitHub, [Online]. Available: https://github.com/leapmotion/LeapMotionCoreAssets [Accessed 2 July 2018].
[76] Technologies, U. (2018). Unity - Manual: Colliders. [online] Docs.unity3d.com. Available at: https://docs.unity3d.com/Manual/CollidersOverview.html [Accessed 1 Aug. 2018].
[77] Docs.microsoft.com. (2018). Introduction to ASP.NET Core. [online] Available at: https://docs.microsoft.com/en-us/aspnet/core/index?view=aspnetcore-2.1 [Accessed 30 Jul. 2018].

[78] Docs.microsoft.com. (2018). Overview - EF Core. [online] Available at: https://docs.microsoft.com/en-us/ef/core/ [Accessed 30 May 2018].
[79] Mysql.com. (2018). MySQL :: MySQL Workbench. [online] Available at: https://www.mysql.com/products/workbench/ [Accessed 30 Jul. 2018].
[80] phpMyAdmin. (2018). phpMyAdmin. [online] Available at: https://www.phpmyadmin.net/ [Accessed 30 Jun. 2018].

Attachments

```
RTIMU *imu;                                                    // the IMU object
RTFusionRTQF fusion;  // the fusion readings (Euler Angles and Quaternions) object
RTQuaternion gravity, rotatedGravity; // quaternions to Gravity and RotatedGravity
RTQuaternion fusedConjugate;          // quaternions to Fused Conjugated readings
RTQuaternion qTemp;              // quaternions to intermediate variable readings
RTVector3 realAccel;            // vector3 to save linear acceleration calculated
float gyro_x, gyro_y, gyro_z;                      // floats to save gyro readings
float accel_x, accel_y, accel_z;          // floats to save accelerometer readings
float mag_x, mag_y, mag_z;                 // floats to save magnetometer readings
float q_w, q_x, q_y, q_z;                   // floats to save quaternions readings
float roll, pitch, yaw;                    // floats to save euler angles readings

void setup(){…}

void loop(){
   // get the latest data if ready yet
   while (imu->IMURead()) {
      // this flush remaining data in case we are falling behind.
      if (++loopCount >= 10)
         continue;
      fusion.newIMUData(imu->getGyro(), imu->getAccel(), imu->getCompass(),
               imu->getTimestamp());
      // get gyro readings
      gyro_x = imu->getGyro().x() * RTMATH_RAD_TO_DEGREE;
      gyro_y = imu->getGyro().y() * RTMATH_RAD_TO_DEGREE;
      gyro_z = imu->getGyro().z() * RTMATH_RAD_TO_DEGREE;
      // get accelerometer readings
      accel_x = imu->getAccel().x() ;
      accel_y = imu->getAccel().y() ;
      accel_z = imu->getAccel().z() ;
      // get accelerometer readings
      mag_x = imu->getCompass().x() ;
      mag_y = imu->getCompass().y() ;
      mag_z = imu->getCompass().z() ;
      // get quaternion base readings
      q_w = fusion.getFusionQPose().scalar();
      q_x = fusion.getFusionQPose().x();
      q_y = fusion.getFusionQPose().y();
      q_z = fusion.getFusionQPose().z();
      // get euler angles readings
      roll = fusion.getFusionPose().x() * RTMATH_RAD_TO_DEGREE;
      pitch = fusion.getFusionPose().y() * RTMATH_RAD_TO_DEGREE;
      yaw = fusion.getFusionPose().z() * RTMATH_RAD_TO_DEGREE;

      // Linear acceleration calculation
      // 1 - gravity rotation and subtraction
      // 1.1 - create the conjugate of the pose
      fusedConjugate = fusion.getFusionQPose().conjugate();

      // 1.2 - now do the rotation - takes two steps with qTemp as the
      // intermediate variable
      qTemp = gravity * fusion.getFusionQPose();
      rotatedGravity = fusedConjugate * qTemp;

      // now adjust the measured accel and change the signs to make sense
      realAccel.setX(-(imu->getAccel().x() - rotatedGravity.x()));
      realAccel.setY(-(imu->getAccel().y() - rotatedGravity.y()));
      realAccel.setZ(-(imu->getAccel().z() - rotatedGravity.z()));
   }
}
```

Code Snippet 1 – Based on RTIMULib [68] shows how obtain the IMU associated data Euler Angles and Linear Acceleration using quaternions. [From Chapter 3.3]

```
// For Each Analog Input Port
int analogPin0 = A0;                             // select the input Analog pin A0
int enable13 = 13;           // enable input Digital pin 13 to reading sensor A0_1
int enable12 = 12;           // enable input Digital pin 12 to reading sensor A0_2
int sensorValue1 = 0;       // variable to store the value coming from sensor A0_1
int sensorValue2 = 0;       // variable to store the value coming from sensor A0_2
int value1_min, value1_max;       // variables to store the values A0_1 to mapping
int value2_min, value2_max;       // variables to store the values A0_2 to mapping
int FlexiForce_A0_1;     // variable to store the value constrain from sensor A0_1
int flexSensor_A0_2;     // variable to store the value constrain from sensor A0_2

void getMultipleAnalogReadings(){
   // read the value from sensor 1:
   digitalWrite(enable13, HIGH);
   sensorValue1 = analogRead(analogPin0);
   FlexiForce_A0_1 = map(sensorValue1, value1_min, value1_max, 0, 100);
   digitalWrite(enable13, LOW);
   delay(2);                                                 // delay milliseconds
   // read the value from sensor 2:
   digitalWrite(enable12, HIGH);
   sensorValue2 = analogRead(analogPin0);
   flexSensor_A0_2 = map(sensorValue2, value2_min, value2_max, 0, 90);
   digitalWrite(enable12, LOW);
   delay(2);                                                 // delay milliseconds
}

void setup(){
   […]
   // For Each Analog Input Port
   pinMode(enable12, INPUT);
   pinMode(enable13, INPUT);
   pinMode(analogPin0, INPUT);
}

void loop(){
   […]
   // Get data of 2 analog sensors for each analog pin
   getMultipleAnalogReadings();
}
```

Code Snippet 2 – Shows how Time Multiplexing is performed to read 2 analog sensors on each analog pin. [From Chapter 3.3]

Appendix A - Article

Article: **Wearable and IoT Technologies Application for Physical Rehabilitation**

This article has been accepted and presented in 1st International Symposium on Sensing and Instrumentation in IoT Era, September 6-7, 2018, Shanghai, China, 2018 IEEE International Instrumentation and Measurement Technology Conference.

ISSI **2018 1st International Symposium on Sensing and Instrumentation in IoT Era**

September 6-7, 2018 - Grand Trustel Purple Mountain Hotel, Shanghai, China

Call for Papers

ISSI 2018 represents a forum where researchers, scientists, engineers and practitioners around the world will promote the advances in sensing and instrumentation for IoT. The symposium valorizes new and original research in IoT applications fields, such as smart city, transportation, healthcare and farming.

The full version papers (4 to 6 pages) submitted to ISSI 2018 will be accepted based on peer review process. The review process may conduct to some revision requests for the final version of the paper that must be mandatory fulfilled to have the paper accepted for the symposium proceedings.

The accepted papers will be published in conference proceedings and will appear in **IEEE Xplore** and **EI Compendex**. Selected papers will be considered for publication in special issue of **Sensors**, SCI open Journal.

Topics of the symposium will include, but are not limited to:

- Smart Sensors and Wireless Sensor Networks for IoT;
- Test and Automated Instrumentation for IoT;
- Localization, Algorithms for Smart Sensor Network;
- Software Platforms and Middleware in Smart IoT Systems;
- IoT in Retail Logistics;
- IoT in Supply Chain Finance and Management;
- Computer Vision and Application in IoT;
- IoT & Wearable Solutions for Healthcare;
- IoT System Design Methodologies;
- Big Data Processing in IoT Systems;
- Energy Harvesting and Scavenging for Wireless Sensors Networks and IoT;
- Education and Training for Networked Instrumentation and Measurement;
- Standards for IoT and IoT Security;
- Sensors and IoT for Smart Ports and Logistics;
- Industrial Internet of Things;
- Sensors and IoT for Structural and Infrastructural Health Monitoring;
- IoT for Healthcare;
- IoT Applications in Smart Cities;
- IoT Applications for Smart Home;
- IoT in Smart Farming;

Many other new initiatives and opportunities to encourage your active participation in the conference are planned, that will make ISSI 2018 a vibrant event to meet with people in sensing and instrumentation field.

Visit the conference website for each specific call and additional news:

http://issi2018.csp.escience.cn

ORGANIZERS

Honorary Chairs

Youfang Huang
Shanghai Maritime University, China
Wei Yan
Shanghai Maritime University, China

General Chairs

Octavian Postolache
ISCTE-IUL & IT, Portugal
Yongsheng Yang
Shanghai Maritime University, China
Xin Wang
Fudan University, China

Technical Program Chairs

Subhas Mukhopadhyay
Macquarie University, Australia
Domenico Capriglione
Università degli Studi di Salerno, Italy
Daqi Zhu
Shanghai Maritime University, China

Publication Chairs

Bin Yang
Shanghai Maritime University, China
Huafeng Wu
Shanghai Maritime University, China
Daofang Chang
Shanghai Maritime University, China

Local Organization Chairs

Chaofeng Li
Shanghai Maritime University, China
Ziyu Lu
Shanghai Pudong Federation of R&D Institutions, China

IMPORTANT DATES

May 11, 2018
Special Session Workshop & Proposals Deadline

June 1, 2018
Paper Submission Deadline

June 25, 2018
Paper Acceptance Notification Deadline

July 15, 2018
Camera-ready Submission & Early Registration Deadline

IEEE INSTRUMENTATION & MEASUREMENT SOCIETY
上海海事大学 Shanghai Maritime University
ISCTE IUL Instituto Universitário de Lisboa
上海市计算机学会 Shanghai Computer Society
instituto de telecomunicações

Wearable and IoT Technologies Application for Physical Rehabilitation

Ricardo Alexandre
Instituto Universitário de Lisboa, ISCTE-IUL
Instituto de Telecomunicações, IT-IUL,
Lisbon, Portugal
rjfae20@gmail.com

Octavian Postolache
Instituto Universitário de Lisboa, ISCTE-IUL Instituto de Telecomunicações, IT-IUL, Lisbon , Portugal
opostolache@lx.it.pt

***Abstract* — This paper presents a set of smart gloves to be used for natural interactions with therapeutic serious games for upper limb rehabilitation. These games provide to the patients with motor impairments the possibility to perform exercises in a highly interactive way based on different VR scenarios. The system is composed by a set of software tools that support the collection and processing of data as so as a set of highly interactive therapeutic games to include in hand and finger rehabilitation session exercises. The acquired data help the physiotherapist to evaluate the patients' performance throughout the different physical activities during a rehabilitation session. The data acquisition is fulfilled by an Arduino Nano computation platform connected to the analog measurement channels materialized by piezo-resistive force sensors. The data communication between the smart sensors and the VR computation platform is performed using Bluetooth wireless communication protocol. Some metrics associated with the upper limb training were also developed to evaluate the patient motion during the training session. Several tests and experimental results are included in the paper.**

Keywords — *Internet of Things; Wearable Sensors; Physical Rehabilitation; Pervasive Healthcare; Wearable Wireless Sensor Network; Therapeutic Serious Games; Force Sensing, Motion Sensing.*

I. INTRODUCTION

From a few decades to this part, the main socio-demographic trends are defined by the increase of the percentage of population living in the cities as well as the aging of the population itself. According to some studies, by 2050, the number of elderly people on the planet will exceed that of young people for the first time in human history [1], and about 70% of the world population is expected to reside in urban areas [2]. These trends make necessary to integrate new technologies and innovative solutions that respond to new environmental, social and health challenges in order to offer new services that can optimize the quality of life cities.

The Internet of Things (IoT) is a contemporary technology with a great development in the century XXI to be part of new healthcare services that will improve quality of life and reduce the costs. The tendency to assure IoT compatibility of the medical devices improves the quality and effectiveness of the service, assuring continuous healthcare monitoring of elderly, patients and patients with chronic disease. The latest developments in wireless sensors networks and cloud technology are providing solutions that can transform our cities into more connected, sustainable and collaborative environments. Recently, wearable devices are emerging and forming a new concept – Wearable IoT (WIoT) because of its ability to detect, transmit and analyze the data. Some development platforms, like Kaa Project [3] have emerged to complements wearable technology with superb ready-to-use IoT functions and applications. The next generations of WIoT promise to revolutionize the healthcare industry, where individuals are easily monitored by portable sensors for personalized health and wellness information - vital body parameters, physical activity, behavior and other critical parameters that affect the quality of everyday life [4]. These technologies are designed to provide better physiotherapist/patient communication, to deliver more efficient care and enable remote patient monitoring as part of independent living solutions.

A objective evaluation based on the measurement of motor activity parameters, especially the functionality of the upper and lower limbs during the rehabilitation period, offers the possibility to increase patients' motivation through adequate feedback and also provides data associated with the physical rehabilitation reports that can be used as the starting point for improved communication between physiotherapists and rehabilitation procedures implemented, but also to improve communication between the physiotherapist and the user of rehabilitation services. Good therapeutic communication may imply a greater ability to obtain valid informed consent, positive clinical outcomes, higher levels of patient satisfaction, higher levels of patients' compliance with rehabilitation programs, lower levels of patient frustration [5].

On the other hand, the health community has shown a great interest in therapeutic approaches based on computer therapeutic games. The concept of therapeutic games refers to the use of computer games without the primary purpose of providing pure entertainment [6]. Virtual reality (VR) implementation for training exercises in physiotherapy interventions has shown important advantages both as technique for objective assessment of treatment as well for engagement of patient in healthcare process [7]. According to the experience described [8], the integration of VR technology with therapeutic games provides greater motivation and more motivation during its rehabilitation activities.

In order to bring these technologies closer to the Health and Physical Rehabilitation Industry, several studies were carried out. In [9], the authors presented the initial design of a low-cost hardware and software platform developed with the integration of a user-friendly graphical user interface (GUI) which facilitates the real-time acquisition of ground reaction force,

978-1-5386-5638-9/18/$31.00 ©2018 IEEE

acceleration and direction of feet, and automatic detection representation of gait impairments. The authors focused their efforts on studying and developing a hardware and software system capable of evaluating the gait process of a human being. They use a prototype of a shoe, equipped with sensors for strength, direction and acceleration, as well as a Bluetooth transceiver of very low consumption. The data coming from the star topology implemented with two end nodes and a coordinator characterized by XBee Wireless communication and USB port connected to a host Laptop computer connected to the internet are processed to extract information about normal and abnormal gait.

Carvalho et al presents in [10] an overview of the technologies and results of integration and testing of e-textile sensors monitoring of biological signals (ECG - electrocardiogram and EMG - electromyography) and environmental conditions (humidity). A seamless jacquard sewing machine was used to integrate these sensors. The integration of the elements (sensors and connections) in the Clothing presents great advantages assuring physical comfort and the psychological comfort of the user.

In [11] are presented a computer therapeutic game structure using a natural user interface (NUI) Kinnect. The platform provides to the user a set of 3D Virtual Reality (VR) scenarios for upper limb rehabilitation. Several metrics are calculated to extract information about the training outcome. Measurements are carried out using a set of portable inertial measurement units (IMUs) attached to the upper limbs. The framework is designed to be part of a remote physiotherapy service that can improve the physical rehabilitation process. In [12] an identical system is presented that uses a Kinnect Natural User Interface, with a set of therapeutic games focused on performing physical rehabilitation exercises, both upper and lower limbs.

The work intends to create new type of user interaction during training based on Therapeutic Games. Two smart gloves that include IMU and force sensors were developed to be used for the interactions with the serious games scenarios for upper limb rehabilitation. These wearable devices equipped with smart sensors, are connected to a user-friendly graphical interface (connected to a Cloud database). The smart gloves provide the applied forces measurement during the rehabilitation of hands and fingers. In parallel, the doctor responsible for the physiotherapeutic session can formulate a more assertive and correct diagnosis about the patient's condition and its evolution, with the data extracted from the patient's use of the platform.

II. System Description

The system architecture includes a low-cost hardware and software including a set of software tools, highly interactive therapeutic games supported by wearable smart sensors and virtual reality. It materializes a natural interface technology for the physical rehabilitation of hands and fingers, applied to users characterized by physical and motor limitations. The system also includes a headband based on same electronic circuit for measuring rotation angles of the head with the aim of integrating the patient into a First-Person Controller during the exercises of the therapeutic games.

This system intends to present new metrics to assess the patient's rehabilitation status and to improve their physical performance complementing the traditional rehabilitation systems in the physiotherapy area. The system block diagram is presented in Figure 1.

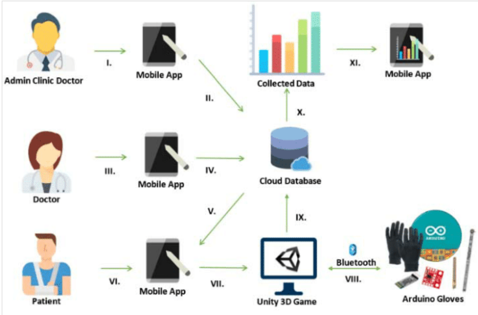

Fig. 1. – The therapeutic serious game system block diagram.

In Figure 1 are considered:

I.- Create and Update Doctor Profiles; **II.**- Settings sent to the Cloud Database; **III.**- Create and Update Patient Profiles; Selection of the game and configuration of the parameters to evaluate by Doctor; See all results of tasks; **IV.**- Settings sent to the Cloud Database; **V.**- Database sends the necessary data for the configuration of the game; **VI.**- Use app to access Unity Games via QRCODE, with all parameters set; See all results of tasks. **VII.**- Play the Therapeutic Games with Gloves; **VIII.**- Arduino collects sensor data, send to game via Bluetooth Module; **IX.**- The data collected from the game is stored in the Cloud Database; **X.**- All data collected are processed to assist a better medical report; **XI.**- The doctor gives a report to the Patient via app.

The implemented IoT Architecture block diagram is presented in Figure 2.

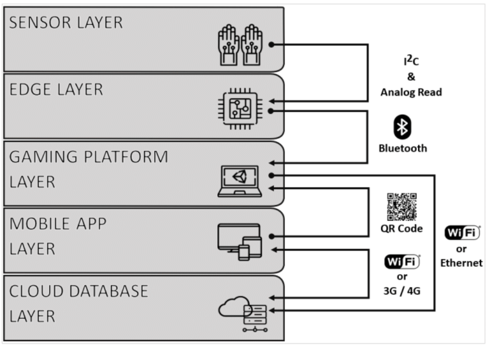

Fig. 2. – The IoT architecture block diagram

In Figure 2 the following topics are considered:

- The Sensor Layer is composed of the IMUs (MPU-9250) and the analog sensors (FlexSensors 2.2" and FlexiForce A201) applied to both gloves.

- The Edge Layer, composed of Arduino Nano microcontrollers, after calibration, acquires IMU data, through I²C (Inter-Integrated Circuit) and analog sensors through the *AnalogRead* method. This data is sent by Bluetooth protocol to Gaming Platform Layer.

- Data generated by the Game itself, that is, scores, angles of movement, or feedback force fingers as well as those that arrive at the Gaming Platform Layer, via Bluetooth are processed and saved on the Internet, in a cloud server in the Cloud Database Layer. This process requires a Wi-Fi or Ethernet connection.

- The Cloud Database Layer will store all data, from the user profile accounts created by the Admin 'system or by the medical staff in the Mobile Application, as well as all exercise monitoring data from the Gaming Platform.

- To access the Game profile, for each patient, there will be a Mobile App Layer, responsible for performing profile validation on the Gaming Platform, through a unique QR Code generated, for each patient. This QR Code will contain all the information necessary to configure the Game and the parameters to be evaluated by the physiotherapist in each training session. This Layer will be responsible for the analysis, processing and visualization of all the data. This process requires a bidirectional Wi-Fi, 3G or 4G connection.

A. *Hardware – Smart Gloves*

The data acquisition is performed using Arduino Nano microcontroller due to its higher autonomy (approximately 60 mAh) and reduced dimensions (approximately 20mm x 35mm).

SparkFun MPU-9250 IMU Breakout was used as the system IMU. It combines two chips: the MPU-6500, which contains a 3-axis gyroscope as well as a 3-axis accelerometer, and the AK8963, which features a 3-axis magnetometer. The MPU-9250 uses 16-bit analog-to-digital converters (ADCs) for digitizing all 9 axes [13]. By merging the data from these sensors, you can obtain reliable values for the rotational movements of the real objects by entering those values in the actions on the virtual objects.

The acquisition of the information associated with the applied force on objects, during the fingers' movements is based on a set of sensors Flexi Force A201 [14] that is included in the system. The sensors are mounted on the glove level, one for each finger. The finger flexion and compression is extracted using Flex Sensors 2.2" [15] are used, also one for each finger. During the game, the finger angular values and tips finger force are visualized on the VR developed in Unity 3D. To be able to read 10 analog output sensors using Arduino Nano platform, a time multiplexing is performed on each of the analogue input, so that it is possible to obtain reliable readings of the two analog sensors associated to each finger using a single analog input channel.

Data acquired from the sensors are primary processed and sent through Bluetooth using a HC-05 module previously configured as a slave, to a COM serial port of the client platform where the Serious Therapy Game developed in Unity 3D runs. Figure 3 shows the set of sensors that are used for each glove.

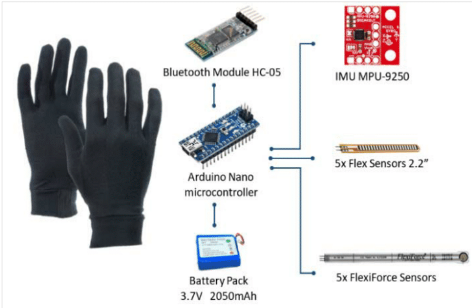

Fig. 3. –Smart gloves: sensing and computation set.

Figure 4 presents the implemented smart glove prototype, including the force sensors placed on the fingers, on top of the glove (only for perception) and the signal conditioning, primary processing and communication board.

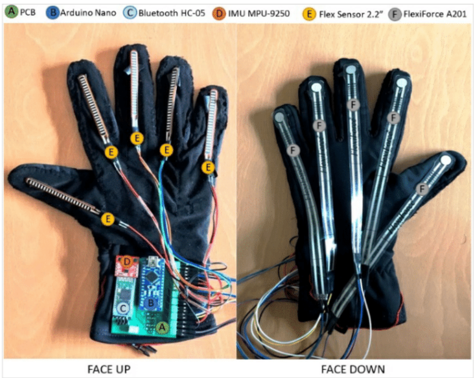

Fig. 4. – Hardware components attached on a glove.

B. *Serious Game Desciption*

With the goal of making physiotherapy exercises more accessible and highly interactive with the user, several Virtual Reality scenarios for the Gaming Platform are designed. Thus, the implemented system permits to animate an Avatar using a First-Person Controller. This controller method allows the game scene Camera to be placed in the position of the Avatar's eyes. On Classic PC Games, that use First-Person Controller this player construction process is usually used with the mouse cursor, where the rotation inputs of the Avatar are generated by the cursor movements. To avoid using the mouse cursor during the game, we will use an electric circuit like the one used in the gloves, which will not include analog sensors, just an IMU. A headband will be responsible for collecting the head rotation value of the patient, which will be used as the rotation input of the player's avatar head.

The process of serial ports reading within the game software is done with threads, as the system reads data from 3 different devices (2 devices of the smart gloves) 1 device mounted on the head at the same time, allowing a fluid interaction with the Virtual environment. Figure 5 shows one scene of the Game.

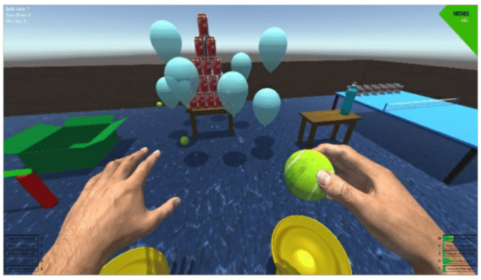

Fig. 5. – Game Scene for "Cans Down Game" developed in Unity 3D.

In this Game is possible grab some objects like balls and pick up objects, and place it on a specific box or throwing to cans or cups. In real-time, we can see the feedback force, on coloured corner bars.

The data obtained from the game platform are submitted to a remote server database. At the end of the session, the data results are sent to the server at same way, and processed to extract values of training effectiveness.

The physiotherapist has the possibility to monitor the results obtained by the patient during the training based on serious game extracting information about the upper limb motor capabilities.

III. EMBEDDED SOFTWARE

The usage of MEMS IMU [16] including accelerometers, gyroscopes and magnetometers can be used to obtain accurate measurements through sensor fusion algorithms. MEMS IMUs are vulnerable to various kinds of noise and errors. Accelerometers are extremely sensitive to attitude changing and impact forces while gyroscopes are sensitive to temperature changes and suffer from a slow-changing bias. To summarize, accelerometers have poor dynamic features and gyroscopes have poor static features. Therefore, an Attitude and Heading Reference System (AHRS) algorithm is needed to fuse the data from different sensors to overcome the drawbacks of each of them and take the most reliable part from them respectively to give a best prediction of the sensor actual status.

AHRS algorithm is the foundation of position estimation [17] because gravity must be removed completely from the accelerometer to get linear acceleration. Only then can the integration be done without being concerned about the drift. Otherwise small errors in the measurement of acceleration are integrated into progressively larger errors in velocity, which are compounded into still greater errors in position. Since the new position is calculated from the previous calculated position and the measured acceleration and angular velocity, these errors accumulate roughly proportionally to the time since the initial position was input.

Like the accelerometer, gyroscope also has the integration drift problem-small errors in the measurements are dramatically amplified by integration over time. However, gyroscopes suffer from another kind of drift. Gyroscopes are sensitive to changes in temperature, which brings about slow-changing deviations just like integration drift did for the accelerometer [18].

The first thing needed is the roll and pitch information. The convention used, if the device is an aircraft for example, then the x-axis points forward from the nose as seen by the pilot, the y-axis points out along the right-wing and the z-axis points straight down. So, a positive roll angle means that the right-wing rolls down, a positive pitch angle means that the nose points up and a positive yaw angle means that the nose rotates towards the right.

In this case, for the implementation of an AHRS algorithm, we base the sketches in the available libraries for a set of many IMUs at richardstechnotes' github, RTIMULib-Arduino [19] and kriswiner' github library [20]. These libraries support the correction of arbitrary, non-standard, coordinate axis-aligned IMU orientations. However, it is very often true that an installation of an IMU in any device characterized by a few degrees offsets on each axis. The correction is done using quaternions. Tilt compensation is required because the magnetometer reads the magnetic field in the orientation of the object where it is mounted, while the required measurements are in the horizontal plane.

By utilizing the accelerometer output, rotation around the X- axis (roll) and around the Y-axis (pitch) can be calculated. If Accel_X, Accel_Y, and Accel_Z are accelerometer measurements in the X-, Y- and Z-axes respectively, equations 1 and 2 show how to calculate the pitch and roll angles (in radians):

$$Pitch = arctan\left(\frac{Accel\_X}{(Accel\_X)^2 + (Accel\_Z)^2}\right) \qquad (1)$$

$$Roll = arctan\left(\frac{Accel\_Y}{(Accel\_Y)^2 + (Accel\_Z)^2}\right) \qquad (2)$$

In order to measure rotation around the Z-axis (yaw), the other sensors need to be incorporated with the accelerometer. The following flow chart shows how it's do it using quaternions.

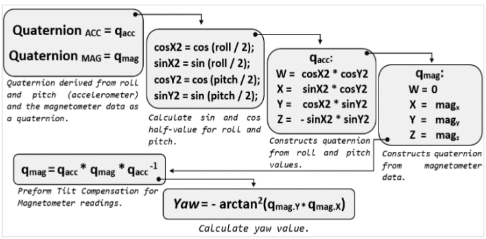

Fig. 6. – Flow chart showing how calculate Yaw using quaternions.

```
RTIMU *imu;                                          // the IMU object
RTFusionRTQF fusion;                              // the fusion object
RTIMUSettings settings;                         // the settings object
RTVector3 realAccel;
RTQuaternion gravity, rotatedGravity;
RTQuaternion fusedConjugate;
RTQuaternion qTemp;

void setup(){…}

void loop()
{
    // get the latest data if ready yet
    while (imu->IMURead()) {
      // this flushes remaining data in case we are falling
      // behind.
      if (++loopCount >= 10)
          continue;
      fusion.newIMUData(
          imu->getGyro(),
          imu->getAccel(),
          imu->getCompass(),
          imu->getTimestamp()
      );

      // [1.] gravity rotation and subtraction
      // [1.1.] create the conjugate of the pose
      fusedConjugate = fusion.getFusionQPose().conjugate();

      // [1.2.] now do the rotation - takes two steps
      // with qTemp as the intermediate variable
      qTemp = gravity * fusion.getFusionQPose();
      rotatedGravity = fusedConjugate * qTemp;

      // now adjust the measured accel and change the signs to make
      // sense
      realAccel.setX(-(imu->getAccel().x() - rotatedGravity.x()));
      realAccel.setY(-(imu->getAccel().y() - rotatedGravity.y()));
      realAccel.setZ(-(imu->getAccel().z() - rotatedGravity.z()));
    }
}
```

Snippet Code 1. – Extracted from RTIMULib showing how calculate Linear acceleration using quaternions.

The first part of the Scheme [Fig. 6.] creates a quaternion *q* representing the roll and the pitch - all that stuff with sin and cos is how you create a quaternion from Euler angles (yaw is always 0 for this purpose so this code has been optimized to reflect that). Next, the magnetometer vector is placed in a quaternion, ready to be tilt compensated. In the line before the last, the magnetometer information is rotated by the quaternion from the roll and the pitch and becomes the tilt-compensated version (that is, the values you would get if the magnetometer were perfectly horizontal). Lastly standard way of generating yaw value is done from tilt-compensated magnetometer readings. Using the same principle, it is then possible to calculate the Linear Acceleration (or residual, without gravity component). The snippet code above [1.] taken from [19] shows how to calculate Linear Acceleration using quaternions.

The double integration of residual accelerations with respect to time to give a change in position, its simple in principle, but is not like that. Accelerations can be mathematically integrated once to obtain velocity and twice to obtain changes in position. Only the accelerations due to effective hand motion must be integrated. Since the accelerometer provides a combination of accelerations due to motion, gravity and error, it becomes necessary to separate these acceleration components. Errors in measured accelerations can be deterministic (bias, scale factor and axis misalignment) and/or random errors. Such errors propagate through the integration process, causing a considerable drift in estimated positions. Owing to drift error, position estimation cannot be performed with adequate accuracy for periods longer than few seconds.

IV. Results and Discussions

The preliminary tests with the serious game implemented were performed in an ISCTE-IUL classroom with two young male healthy volunteers (25 years, 26 years), height (1.78 m and 1.75 m), respectively. For the "Cans Down Game" from Unity 3D platform it is possible to grab virtual objects (tennis balls) and toss them towards the stack of cans. To perform this test, we will only consider the data collected from the glove for the right hand, because the prototype is still in development. The results obtained are presented in the following charts:

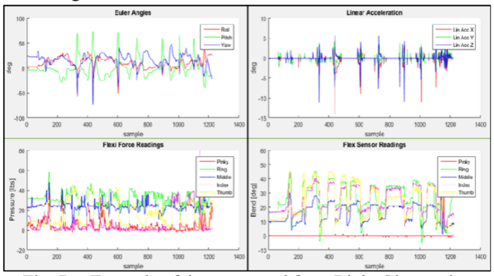

Fig. 7. – Example of data extracted from Right Glove using MatLAB for participant 1.

The exercise of participant 1 lasted for 85 seconds. Through the charts presented in Fig. 7, it is possible to observe, through the Euler angles variations that 7 attempts were made to strike the virtual balls, as there were peaks in the Pitch value. It is also possible to observe all the variations of Linear Acceleration in the 3 Cartesian components, where there is a greater variation in the Z-axis. The Flexi Force readings allows us to observe all the forces applied at the tips of each finger, in the movements of opening and closing the hand, where it can be verified that Index Finger was the one that captured higher values during the exercise. This value is estimated for a range of 0-100 lbs. The Flex Sensor readings chart allows us to visualize all the angular flexures performed during the exercise, the values being estimated for a range of 0-90 degrees.

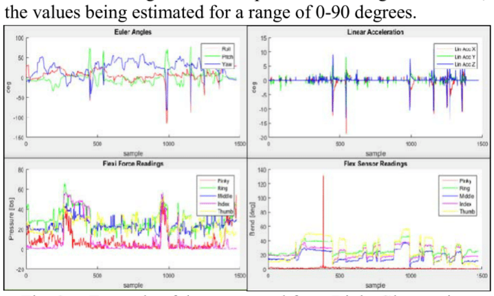

Fig. 8. – Example of data extracted from Right Glove using MatLAB for participant 2.

Fig. 8 presents the exercise performed by participant 2 (103 seconds duration). The charts present the Euler angles variations for 6 attempts that were made to strike the virtual balls, highlighted by the evolution of Pitch value.

In both tests it was verified that the reading value collected from the FlexSensor, referring to the pinkie finger is practically null because the sensor has been damaged. In the future will be considered the replacement of it.

Based on data associated with serious game session the physiotherapist will be able to have a better insight in how patients are progressing during sessions performed at home or in clinics, and they can perform a better evaluation of motor capability status of their patients adapting the physical rehabilitation plan to the patient needs.

V. Conclusions and Future Work

The main purpose of this paper is to describe a serious therapeutic gaming system that combines wearable sensors and VR scenarios. The used technology for the VR component was Unity 3D different tests being carried out. The developed system, is characterized by an IoT architecture including force and IMU sensors deployed on the glove and connected to a microcontroller as edge computation platform. The system allows to evaluate the movements of the hands and fingers in real time, providing information to the physiotherapist about the correct execution of the exercises. Using this system, the physiotherapist can follow the patient during the prescribed training plan for days, weeks or months. Taking into account the recorded data associated with the patients the data analytics can be applied to extract the rehabilitation model and to predict the rehabilitation outcome.

The mobility characteristics of the system allows the patient to stay in his home and to perform the exercises proposed by the physiotherapist. Easy installation of the system and calibration of the device by non-specialists for use in a non-clinical environment, make this solution suitable for use at home. Patients feel psychologically better in their own environment and also the speed of rehabilitation is improved.

The future work is expressed by the possibility to apply Big Data Analysis for the data obtained with developed system and to develop a set of personalized rehabilitation models. In addition, an improvement of the own Virtual environments would make the exercises even more interactive for those who use the system. New games for lower limb rehabilitation will be considered in the future.

Acknowledgment

This research is supported by Instituto de Telecomunicações (IT-IUL) at ISCTE-IUL, Lisbon, Portugal and Fundação para Ciência e Tecnologia (FCT) project PTDC/DTPDES/6776/2014.

References

[1] [Online] Envelhecimento report-online at https://www.unric.org/pt/envelhecimento (Accessed 3 June, 2018)

[2] [Online] WHO MedXForce report-on-line at https://medxforce.com/stroke/Statistics/ (Accessed 3 June, 2018)

[3] [Online] Kaa Project Official Website at https://www.kaaproject.org/wearables/ (Accessed 3 June, 2018)

[4] S. Hiremath, G. Yang and K. Mankodiya, "Wearable Internet of Things: Concept, architectural components and promises for person-centered healthcare," 2014 4th International Conference on Wireless Mobile Communication and Healthcare - Transforming Healthcare Through Innovations in Mobile and Wireless Technologies (MOBIHEALTH), Athens, 2014, pp. 304-307.)

[5] O. Postolache and P. S. Girão, "Physiotherapy smart connected devices for S-health," 2016 7th International Conference on Information, Intelligence, Systems & Applications (IISA), Chalkidiki, 2016, pp. 1-6.

[6] C. Watters, S. Oore, M. Shepherd, A. Abouzied, A. Cox, M. Kellar, H. Kharrazi, F. Liu, A. Otley, "Extending the Use of Games in Health Care", *Proc. 39th Hawaii International Conference on System Sciences (HICSS)*, January2006.

[7] N. Suttiea, S. Louchart, T. Lim, A. Macvean, W. Westera, D. Brown, D. Djaouti, "Introducing the "Serious Games Mechanics. A Theoretical Framework to Analyse Relationships Between Game and Pedagogical Aspects of Serious Games", Procedia Computer Science, Vol 15, pp 314–315, 2012.

[8] Chang, C-Y.; Lange, B.; Zhang, M.; Koenig, S.; Requejo, N.; Sawchuk, A.A.; Rizzo, A.A.; Towards Pervasive Physical Rehabilitation using Microsoft Kinect, *Proc. 6th International Conference on Pervasive Computing Technologies for Healthcare* (PervasiveHealth), San Diego – CA, USA, May 2012.

[9] O. Postolache, P. S. Girão, J. M. D. Pereira and G. Postolache, "Wearable system for gait assessment during physical rehabilitation process," *2015 9th International Symposium on Advanced Topics in Electrical Engineering (ATEE)*, Bucharest, 2015, pp. 321-326.

[10] H. Carvalho, A. P. Catarino, A. Rocha and O. Postolache, "Health monitoring using textile sensors and electrodes: An overview and integration of technologies," *2014 IEEE International Symposium on Medical Measurements and Applications (MeMeA)*, Lisboa, 2014, pp. 1-6.

[11] V. Viegas, O. Postolache, J. M. D. Pereira and P. M. S. Girão, "NUI therapeutic serious games with metrics validation based on wearable devices," *2016 IEEE International Instrumentation and Measurement Technology Conference Proceedings*, Taipei, 2016, pp. 1-6.

[12] D. Ferreira, R. Oliveira and O. Postolache, "Physical rehabilitation based on kinect serious games," 2017 Eleventh International Conference on Sensing Technology (ICST), Sydney, NSW, 2017, pp. 1-6.

[13] [Online] Official Github SparkFun IMU MPU-9250 Breakout at https://github.com/sparkfun/MPU-9250_Breakout (Accessed 11 June, 2018)

[14] [Online] Official Website FlexiForce A201 at https://www.tekscan.com/products-solutions/force-sensors/a201 (Accessed 11 June, 2018)

[15] [Online] Official Website Flex Sensors at http://www.spectrasymbol.com/product/flex-sensors/ (Accessed 11 June, 2018)

[16] Kim A., Golnaraghi M. Initial calibration of an inertial measurement unit using an optical position tracking system; Proceedings of the Position Location and Navigation Symposium (PLANS 2004); Monterey, CA, USA. 26–29 April 2004.

[17] [Online] Attitude and Heading Reference System - Wikipedia Website https://en.wikipedia.org/wiki/Attitude_and_heading_reference_system (Accessed 14 June, 2018)

[18] Zhi, Ruoyu, "A Drift Eliminated Attitude & Position Estimation Algorithm In 3D" (2016). *Graduate College Dissertations and Theses*. 450. [Online] https://scholarworks.uvm.edu/graddis/450 (Accessed 15 May, 2017)

[19] [Online] RTIMULib-Arduino Library Github Webite at https://github.com/richardstechnotes/RTIMULib-Arduino (Accessed 18 June, 2018)

[20] [Online] MPU-9250 Generic Library Github Webite at https://github.com/kriswiner/MPU9250 (Accessed 18 June, 2018)

IEEE

IEEE INSTRUMENTATION & MEASUREMENT SOCIETY

Student Best Paper Award

In the topic of "*IoT for Healthcare*"

At 2018 International Symposium on Sensing and Instrumentation in IoT Era (ISSI2018) to

Ricardo Alexandre and Octavian Adrian Postolache

For the paper entitled

"Wearable and IoT Technologies Application for Physical Rehabilitation"

6 - 7 September 2018, Shanghai, China

Yongsheng Yang, General Chair, ISSI2018

Appendix B – User Manual

ISCTE IUL
Instituto Universitário de Lisboa

DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY

User Manual

Physio Wear - Wearable and IoT Technologies
Application for Physical Rehabilitation

Ricardo José Farinha Alexandre

Advisors:

Professor Octavian Adrian Postolache, Assistant Professor, ISCTE-IUL

Doutor Vítor Viegas, Assistant Professor, Portuguese Naval Academy

October 2018

Figures

Contents

User Manual

This manual aims to present the features available in the Physio Wear's project applications and explain how they work. Chapter 1 is intended to demonstrate the functionalities of the Physio Wear application for the computer, which is intended for patients in the physiotherapy clinic or home. Chapter 2 demonstrates the functionalities of the mobile application, which serves as a support to the clinic manager, enabling the management of its physiotherapists and physiotherapists to manage patients. This application is also intended for patients to access their profile with QR Code of access to the games, as well as the results of the sessions performed. Chapter 3 is intended to explain the method of installation of the above-mentioned applications.

Chapter 1 - Physio Wear Computer Application

1. Overview and Application Description

The Physio Wear Computer Application is intended for patients of physiotherapy clinics, but this manual isn't intended for this type of actor, it should be the responsibility of the physiotherapist to be aware of this manual, to teach and explain to the patient what it is supposed to do.

At the beginning, the user encounters two Introduction splash screens (see Figure 1.1).

Figure 1.1 - Physio Wear Unity Application Introduction splash screens 1 and 2.

After both are finished, a new screen is displayed (see Figure 1.2). This screen is required for the patient to log in so that they can perform their respective exercise plan.

Figure 1.2 - Physio Wear Unity Application LoginQRCode screen.

The patient (with the help of the physiotherapist or someone to supervise) shows in the Webcam of the computer the QRCode of Physio Wear Mobile Application. The application reads the information contained in the QRCode and identifies whether the user is valid. If not, a negative sound is released, and a message is displayed in the screen that the user is invalid. Otherwise, an affirmative sound and a message that the login to the system was successful was released. (see Figure 1.3)

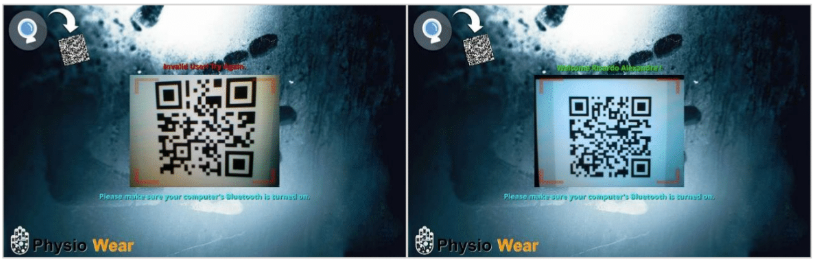

Figure 1.3 - Physio Wear Unity Application Login QR Code screen, showing the messages of invalid user and successful login.

After login, a new screen is loaded with some information about the patient such as name, gender and age (see Figure 1.4), while the system checks for an active exercise plan for the patient.

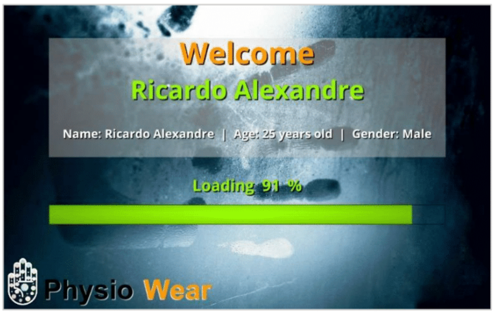

Figure 1.4 - Physio Wear Unity Application Welcome screen.

If there is no exercise plan valid for that day, a new screen is displayed to the patient - No Active Plan Scene - showing an informative message to the patient (see Figure 1.5) and the application restart counter.

Figure 1.5 - Physio Wear Unity Application No Active Plan screen.

If there is a valid exercise plan for the day, the active plan information for that day will be uploaded and a new screen will appear with the summary of the exercise plan and serious game information (see Figure 1.6).

Figure 1.6 - Physio Wear Unity Application Summary Gameplay screen.

Each training plan set up by the physiotherapist allows:

- choose the game to play;
- the start and end dates for which the plan is valid;
- the duration of each session;
- the rest time between sessions;
- the number of repetitions of that plane;
- the members to use;
- the difficulty of the game;
- and some necessary notes;

After the set display time screen has elapsed, a new screen is loaded with calibration instructions and an example image for wearable devices (see Figure 1.7). This step is

fundamental to the game can be performed because it will allow the system to perceive the location of the monitor where the game is running and the position where the user will execute the movements. The Calibration mode is active on 10 first seconds of the Game Scene and can be activated by pressing the **Space Bar** off the computer whenever the user of the serious game so requires. The Game scene starts on when the timer of the display on this screen ends.
It should be during the display of the last 2 screens or in Application Start (depends on how long it takes to do so) that the patient should wear the smart gloves (with help of someone) as well as fix the headband in the most correct place of the head.

Figure 1.7 - Physio Wear Unity Application Calibration screen.

After finishing the time to start the game, the application configures the next game. There are two games available, **Cans Down Game** and **Coffee Pong Game**. The Game Scene is loaded in device calibration mode that ends after finishing the 10-second count displayed on the screen (see Figure 1.8).

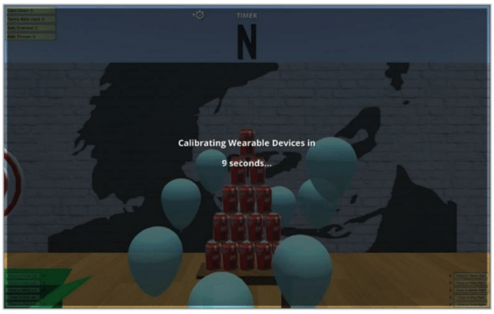

Figure 1.8 - Physio Wear Unity Application Game screen on Calibration Mode on Cans Down Game.

Cans Down Game

The game goal is to knock down the stack of cans by throwing the tennis balls that appear in the golden goblets. To grab a ball, 2 fingers of one hand should touch it simultaneously, it should be the thumb and the other Finger of the hand. To throw the ball, the hand must be open at the end of the movement. A new ball is automatically generated whenever other ball touches the ground. Balloons are a decorative element of the Virtual Reality scenario. For each Can dropped on the floor is gained 1 point. Figure 1.9 illustrates the **Cans Down** game interface.

Figure 1.9 - The Cans Down game interface.

The **T** key when pressed generates new balls at any moment in the game, whenever necessary.

The **U** key when pressed shows the debug information about rotations.

When the patient finishes the game, the results in terms of scores are stored in the database. Three different types of **scores** are stored:

- Total of tennis balls that have been grabbed;
- Total of tennis balls that have been thrown;
- Game score about cans dropped;

Coffee Pong Game

The game aims to get the patient to grab the available ball on the ping-pong table with one hand and pick up the ping-pong paddle with another. The goal is make balls take down coffee cups in front, on another side of table. For each ball thrown, a new ball is placed on table. If the patient drops the paddle to the floor, a new paddle is placed on start position. For each Coffee Cup dropped the patient gains one point. A Figure 1.10 shows an example of the **Coffee Pong** game interface.

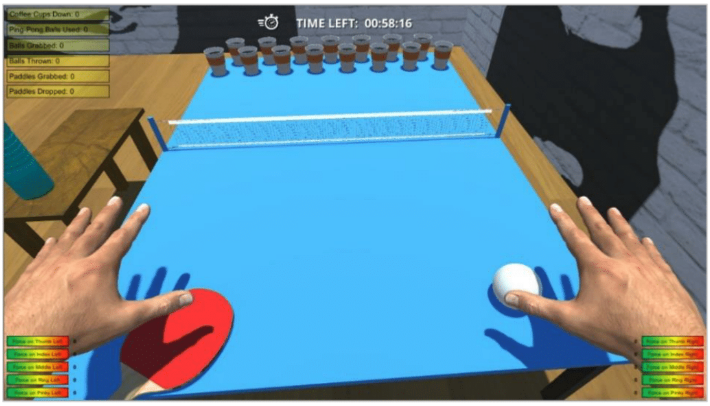

Figure 1.10 - The Coffee Pong game interface.

The **P** key when pressed generates a new paddle and a new ball at any moment in the game, whenever necessary.
The **U** key when pressed shows the debug information about rotations.

When the patient finishes the game, the results in terms of scores are stored in the database. Three different types of **scores** are stored:

- Total of ping-pong balls that have been picked;
- Total of ping-pong balls that have been thrown;
- Game score about coffee cups dropped;

After finishing the game, a screen with the patient's scores is shown (see Figure 1.11).

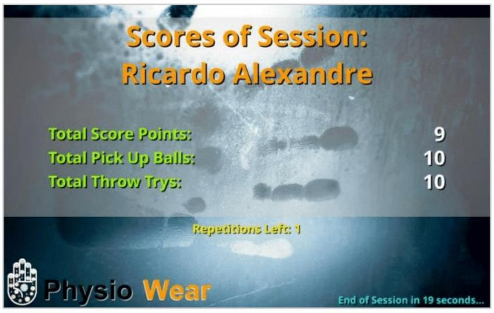

Figure 1.11 - Physio Wear Unity Application Scores screen.

If the patient has repetitions to do regarding the exercise plan, a new screen is loaded, which contemplates a loading bar attached with the rest time defined for the exercise plan, is loaded (see Figure 1.12). After this timeout, the Summary Gameplay Scene is loaded again.

Figure 1.12 - Physio Wear Unity Application Rest Time Scene.

If the patient does not have repetitions to do regarding the exercise plan, anew screen is loaded, which includes a new informative message to the patient (see Figure 1.13). After the screen display time finishes the application terminates.

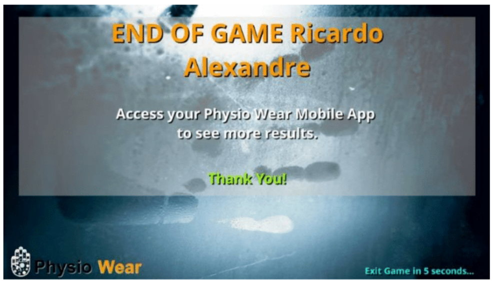

Figure 1.13 - Physio Wear Unity Application End Session Scene.

If the user wishes to restart the application, for example due to an anomaly or to verify that the training isn't the most suitable for the patient, exist the ESC key to exit the application. This method is available in any application window.

**** All images were captured in the built-in application PhysioWearUnity3D-v1.exe created in Unity 2017.3.1f1 (64-bit) and running on Windows 10 Home Edition. ****

Chapter 2 - Physio Wear Mobile Application

1. Overview

There are three types of users in this application, **clinic manager, physiotherapist** and **patient**. Each participant has access to a restricted area in the application, with different functionalities, the clinic manager has features to manage and create new physiotherapists in the system, while physiotherapists have functionalities for patient management, which includes the possibility of creating and managing new training plans, visualizing results and creating new patients in the system. The patient can access the mobile application to check their data in the system, which includes the QR Code of access to the game platform, their exercise plans and the results of the sessions that also include the scores.

When the application is opened, a splash screen is loaded which includes an animation on logo (see Figure 2.1) and then the Login Page is loaded, which will serve all users (see Figure 2.2).

Figure 2.1 - Splash page.

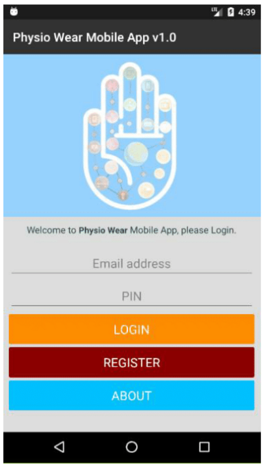

Figure 2.2 - Login page.

For each user to have access to his restricted area in the application, enter his credentials (valid e-mail address and PIN), enter the email in the first field, the PIN in the second field and click the Login button. Figure 2.2 represents the Login Page, that include the area where the user enters the credentials and some others features like logo, Register Button and About Button.

The About Button, is a button that loads the About Page, that includes the information about Xamarin Application Development. This page includes a button linked to Xamarin Official Web Site to see all official documentation (see Figure 2.3).

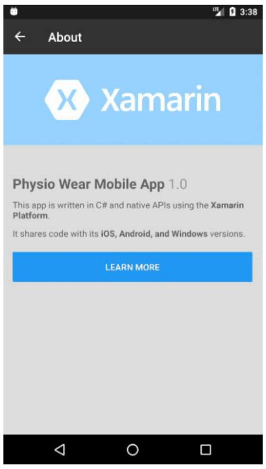

Figure 2.3 - About page.

The Register Button, is a button that serves to register clinics and the corresponding administrator of each clinic. **It should only be used by the Clinic Administrator** and **only at the First Use**, to perform the Clinic Register (see Figure 2.4) and then its Clinic Administrator Profile Register (see Figure 2.5). In this page the Admin fills their information, that includes the valid email address and the PIN to enter in Login Page.

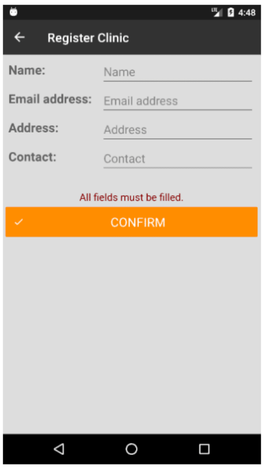

Figure 2.4 - Register Clinic page.

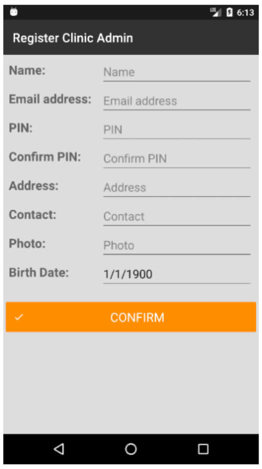

Figure 2.5 - Register Admin Clinic page.

After the registration is done, the administrator can access their content in the application. The features for the Clinic Administrator will then be presented following.

1.1 Admin Clinic Application Mobile Features

After entering the access credentials and logging in (see Figure 2.6), the Home Page is loaded. In the case of the Administrator, the Home Page consists of a button to add Physiotherapist Profiles to the clinic, a Search Bar (to search the Physiotherapists by name) and the List of Physiotherapists (with names in alphabetical order) registered in the respective clinic (see Figure 2.7).

In the top bar of the Home Page, you can see a button to access the Menu Page (see Figure 2.8), a button to refresh the list of Physiotherapists and an application Logout button.

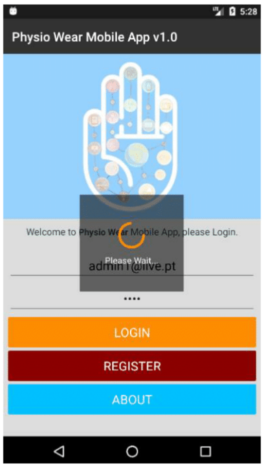

Figure 2.6 - Login page.

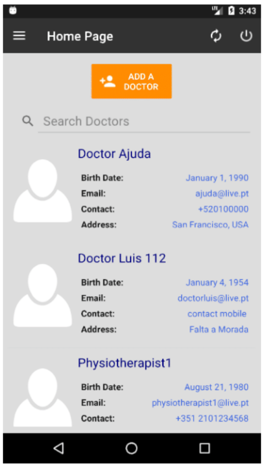

Figure 2.7 - Admin Home page.

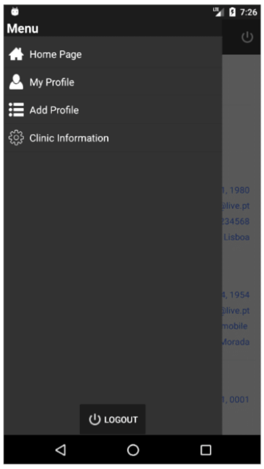

Figure 2.8 - Master Main Menu page.

In Menu Page, the admin can access the main features of the application. This includes access to the My Profile Page (see Figure 2.9) where can view and access the information editing page (see Figure 2.10). It is also possible to access the registration page of new Physiotherapists in the clinic, as in the button on the Home Page (see Figure 2.11), that includes defining the valid email and PIN to Physiotherapists access the mobile application. This menu also includes access to the Clinic Information Page to view all relevant Clinical information (see Figure 2.12). Finally, the Menu page ends with another application logout button.

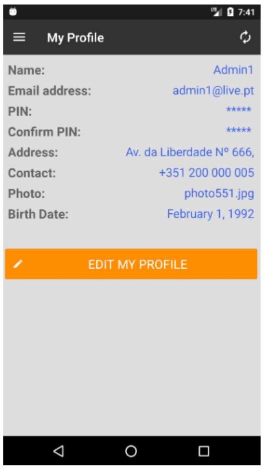

Figure 2.9 - Admin Clinic My Profile page.

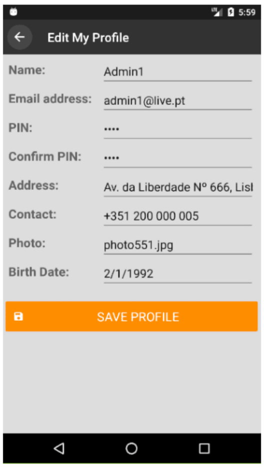

Figure 2.10 - Admin Clinic Edit My Profile page.

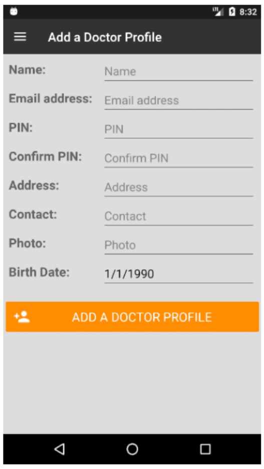

Figure 2.11 - Add Physiotherapist Profile page.

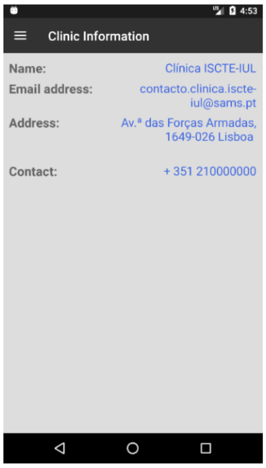

Figure 2.12 - Clinic Information page.

At the end, in their Home Page, the admin clinic can select the Physiotherapist profile from the Physiotherapist Profiles list, view the information (see Figure 2.13), delete the profile in the upper corner or access the profile editing (see Figure 2.14).

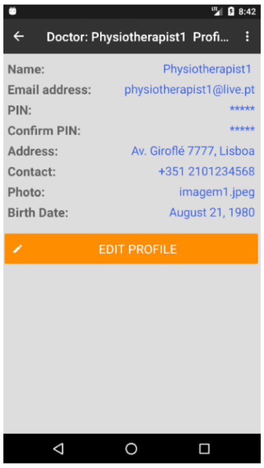

Figure 2.13 - View Physiotherapist Profile page.

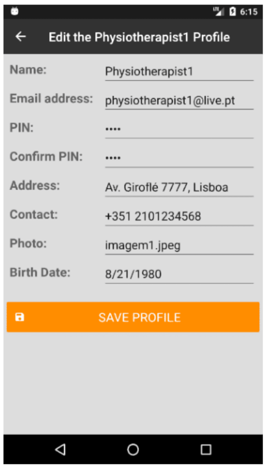

Figure 2.14 - Edit Physiotherapist Profile page.

In this was covered all the main functionalities of the mobile application for the use by the Administrators of the Clinics.

After the registration is done, the Physiotherapists can access their content in the application. The features for the Physiotherapists will then be presented following.

1.2 Physiotherapist Application Mobile Features

After entering the access credentials and logging in (see Figure 2.15), the Home Page is loaded. In the case of the Physiotherapist, the Home Page consists of a set of buttons to see all Patient Profiles, to add Patient Profiles to the clinic and to see only the Favorites Patients profiles, a Search Bar (to search the Patients by name), and the List of Patients associated of Physiotherapist logged (with names in alphabetical order) registered in the respective clinic (see Figure 2.16).
In the top bar of the Home Page, you can see a button to access the Menu Page (see Figure 2.17), a button to refresh the list of Patients and an application Logout button.

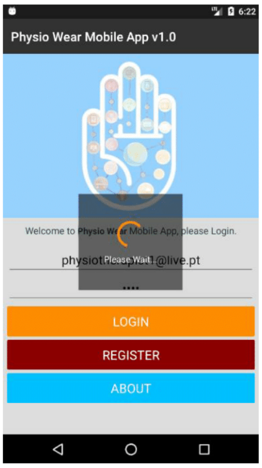

Figure 2.15 - Login page.

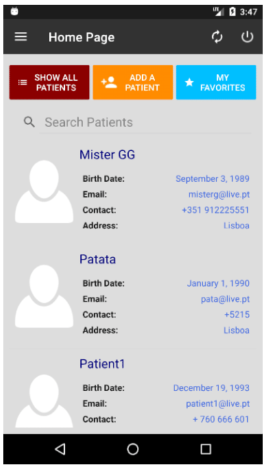

Figure 2.16 - Admin Home page.

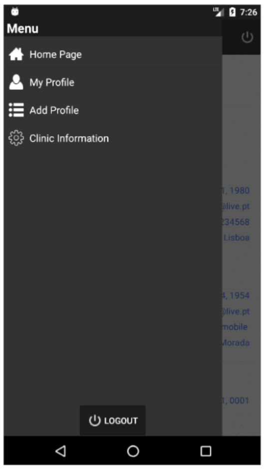

Figure 2.17 - Master Main Menu page.

In Menu Page, the physiotherapist can access the main features of the application. This includes access to the My Profile Page (see Figure 2.18) where can view and access the information editing page (see Figure 2.19). It is also possible to access the registration page of new Patients in the clinic, as in the button on the Home Page (see Figure 2.20), that includes defining the valid email and PIN to Patients access the mobile application.

This menu also includes access to the Clinic Information Page to view all relevant Clinical information (see Figure 2.21). Finally, the Menu page ends with another application logout button.

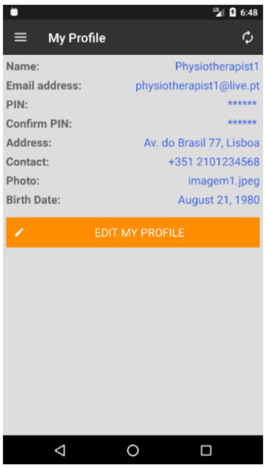

Figure 2.18 - Physiotherapist My Profile page.

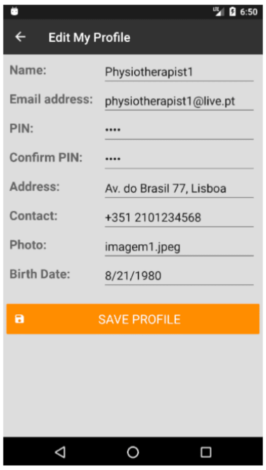

Figure 2.19 - Physiotherapist Edit My Profile page.

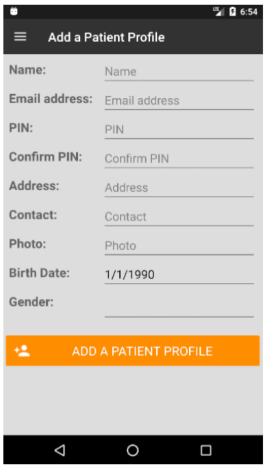

Figure 2.20 - Add Patient Profile page.

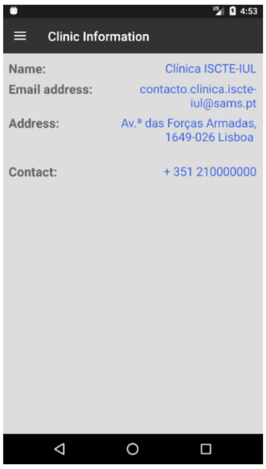

Figure 2.21 - Clinic Information page.

At Home Page it is possible select the profile that we want to see and open it. In View Profiles page (see Figure 2.22) we can see all information about the patients and edit or delete the profiles. The Edit Profiles page is identical of Edit My Profile page (see figure 2.23). When the logged physiotherapist opens a patient profile, is allowed on this page, besides the possibility of changing the patient information, the possibility of including the patient in his favourite list (on right upper corner), as well as visualizing information about the patient's exercise plans (see figure 2.24).

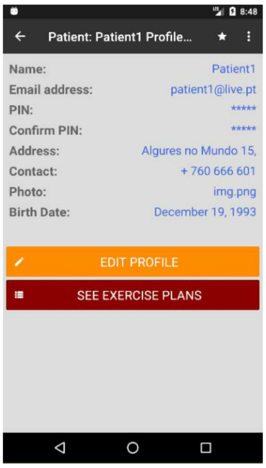

Figure 2.22 - View Patient Profile page.

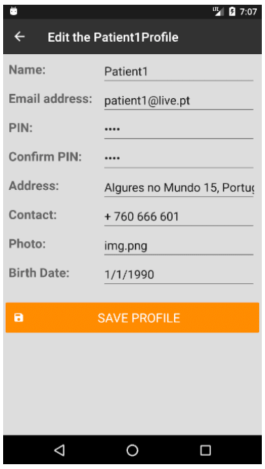

Figure 2.23 - Edit Patient Profile page.

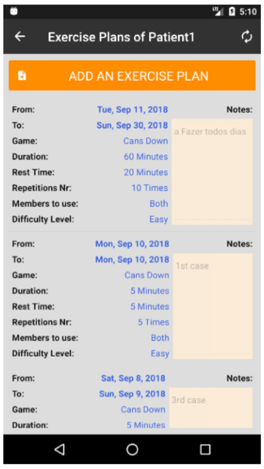

Figure 2.24 - View Exercise Plans page.

When physiotherapist opens the Exercise Plans page, will be loaded a new list of the exercise plans of the patient, if has it, like shown in Figure 2.24. It is possibly to Physiotherapist create a new Exercise Plan (see Figure 2.25) or open an existing one (see Figure 2.26).

· 24 ·

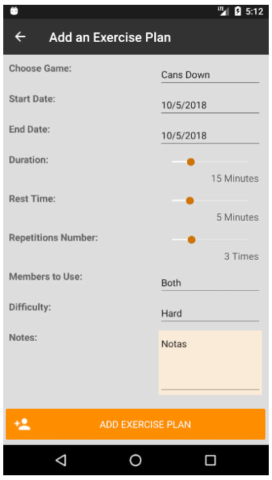

Figure 2.25 - Add Exercise Plan page.

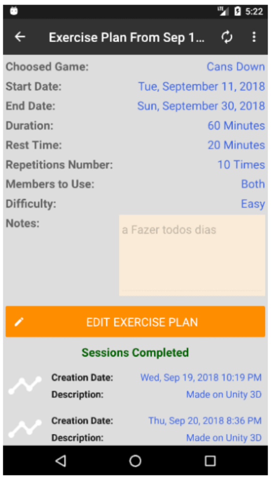

Figure 2.26 - Exercise Plan page.

The Exercise Plan page allows the physiotherapist to edit the selected exercise plan (see Figure 2.27) and visualize a list of all sessions created by patient in game platform to the mentioned exercise plan when plays the games. By selecting one, the Session Report page is loaded to physiotherapists see all results, like collected data of the members used and gaming scores (see Figure 2.28).

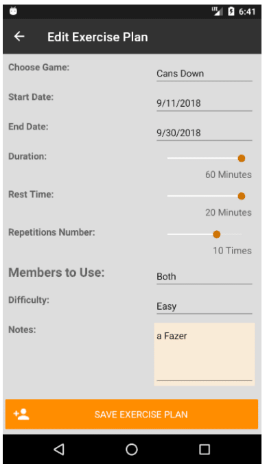

Figure 2.27 - Edit Exercise Plan page.

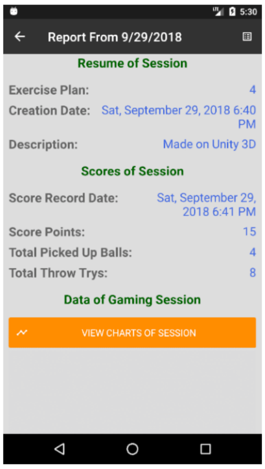

Figure 2.28 - Session Report page.

In this page is also possible see the charts data associated with members and head movements on gaming performance, by clicking the View Charts of Session button. In the charts it is possible select the data to show, by clicking the correspondent button (see Figure 2.29). At the right upper corner of Session Report page exist a button to access the Observation page . The Observation page is for the Physiotherapist to write some notes or topics around the exercise session performed (see Figure 2.30), making it visible to the patient. This information can be changed whenever the Physiotherapist so decides.

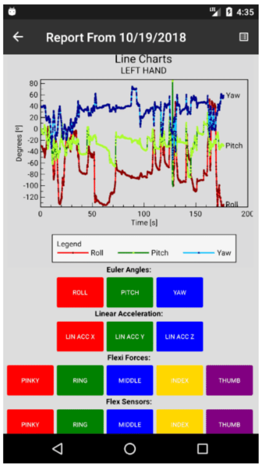

Figure 2.29 - Session Report page showing charts.

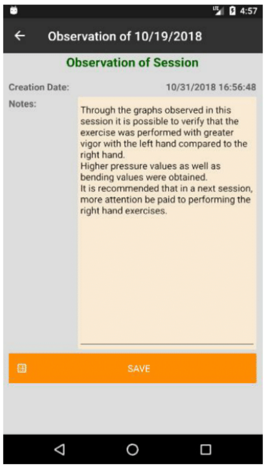

Figure 2.30 - Observation Page created by Physiotherapist.

In this was covered all the main functionalities of the mobile application for the use by the Physiotherapists of the Clinics.
After the registration is done, the Patients can access their content in the application. The features for the Patients will then be presented following.

1.3 Patient Application Mobile Features

After entering the access credentials and logging in (see Figure 2.31), the Home Page is loaded. In the case of the Patient, the Home Page consists of a Search Bar (to search the Exercise Plans by date), and the List of Exercise Plans associated of Patient logged (the list will appear sorted from the most recent to the oldest plan) (see Figure 2.32).
In the top bar of the Home Page, you can see a button to access the Menu Page (see Figure 2.33), a button to refresh the list of Exercise Plans and an application Logout button.

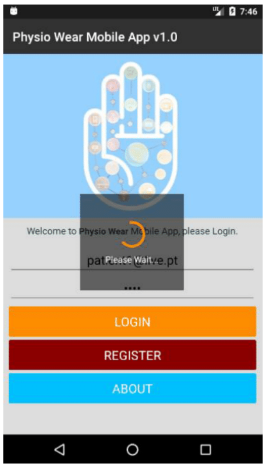

Figure 2.31 - Login page.

Figure 2.32 - Patient Home Page.

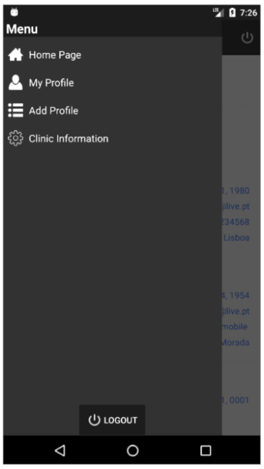

Figure 2.33 - Master Main Menu page.

In Menu Page, the patient can access the main features of the application. This includes access to the My Profile page (see Figure 2.34). The My Profile page shows information about own profile. On My Profile page of Patient there is also a button, which when pressed shows the QR Code associated with the patient's account, which will serve to login to the Game application on Unity 3D platform (see Figure 2.35). It is possible editing the info clicking on Edit Profile button (see Figure 2.36).

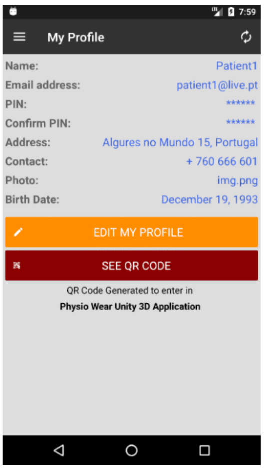

Figure 2.34 - My Profile Patient account page.

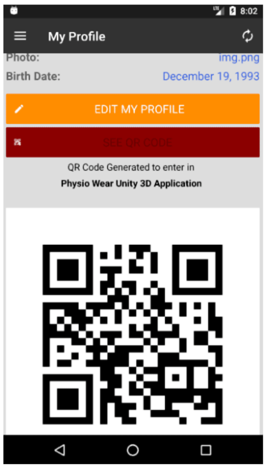

Figure 2.35 - My Profile Patient account page showing QR Code.

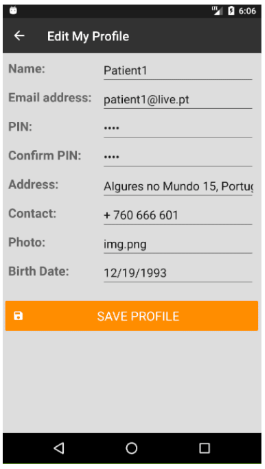

Figure 2.36 - Edit My Profile page.

The Add Profile page is not available for Patient accounts. The menu also includes access to the Clinic Information Page to view all relevant Clinical information and Responsible Physiotherapist information (see Figure 2.37). Finally, the Menu page ends with another application logout button.

Figure 2.37 - Clinic Information Patient Account page.

On the Home page, patients can view the list of all their exercise plans, and open what they want. The Exercise Plan page (see Figure 2.38) allows the patient to see all information about and visualize a list of all sessions created in game platform to the mentioned exercise plan when plays the games. By selecting one, the Session Report page is loaded to physiotherapists see all results, like collected data of the members used and gaming scores (see Figure 2.39). In this page is also possible see the charts data associated with members and head movements on gaming performance, by clicking the View Charts of Session button. In the charts it is possible select the data to show, by clicking the correspondent button (see Figure 2.40).

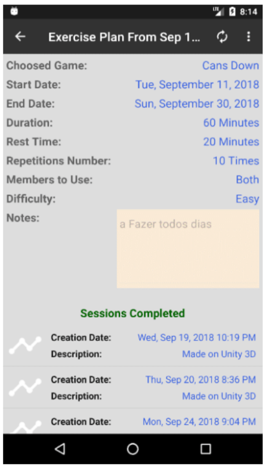

Figure 2.38 - Exercise Plan page on Patient Account.

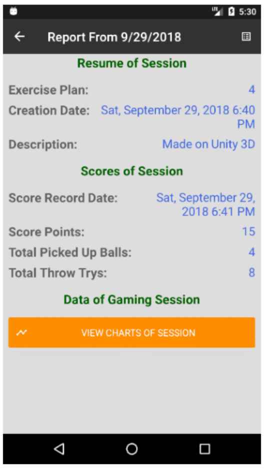

Figure 2.39 - Session Report page.

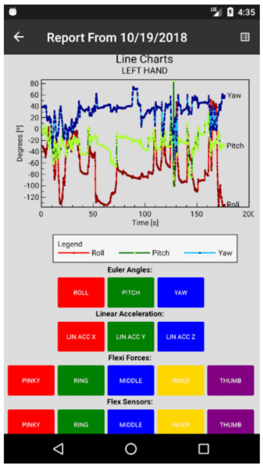

Figure 2.40 - Session Report page showing charts.

At the right upper corner of Session Report page exist a button to access the Observation page. The Observation page is for the Physiotherapist to write some notes or topics around the exercise session performed, making it visible to the patient (see Figure 2.41).

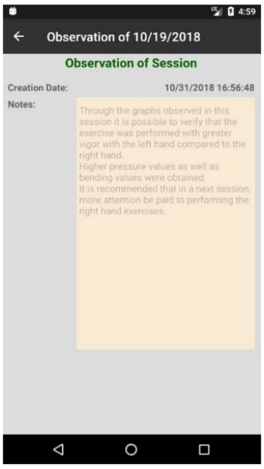

Figure 2.41 - Observation
Page accessed by Patient.

In this was covered all the main functionalities of the mobile application for the use by the Patients of the Clinics.

****All images were captured in the Physio Wear mobile application created in Xamarin.Forms and running on Android OS. ****

Chapter 3 - Installation Manual

1. Overview

This chapter explains how to install the two Physio Wear applications. First, how to install the application for the computer and then the mobile application. Should accesses the pen drive with the contents of Physio Wear, as shown in Figure 3.1.

1.1 Computer Application Installation

The installation of the computer application of the Physio Wear system is simple and is described below.

1. Access the Project Main Folder with all content. Open the "UNITY" folder.
2. Open the file "PhysioWearUnity3D_v1.exe". The graphics game settings window will be launched.
3. When the configuration is done, according to the user, just press the "Play!" button and the Game starts.

It is important to remember that in order to run the game you must first connect the Bluetooth of the system to pair the devices.

1.2 Mobile Application Installation

The installation of the mobile application of the Physio Wear system is simple and is described below.

1. Access the Project Folder with all content. Open the "App + WebAPI" folder.

   <u>In case of Android application:</u>

2. Open the "PhysioWear Android" folder.
3. Connect an Android device to your computer via Bluetooth or USB cable.
4. Enable "Settings> Programmer Options> Allow Unknown Sources".
5. Send the "com.companyname.PhysioWearMobile.apk" to your Android device.
6. Perform the installation and Open it.

In case of Windows (UWP) application:

7. Open the " PhysioWearMobile.UWP" folder.
8. Open the "PhysioWearMobile.UWP_1.0.0.0_x86_x64_arm.appxbundle" on computer and press "Install".
9. When installation is finished open it.

The minimum version of the operating system for the UWP version of the Physio Wear Mobile Application is Windows 10.

1.3 Dressing and Pairing to Wearable Devices

Smart gloves should be worn with resistive piezo sensors placed on each of the appropriate tissue channels as far forward as possible. The sensors on the upper face should be terminated on the nails and the sensors on the underside should be in contact with the fingertips. The electronic device housing should be attached to the wrist area and can be adjusted by the Velcro. The figure 3.1 shows how the smart glove is dressed. The device should be oriented as shown in the figure.

Figure 3.1 - The smart glove for right hand dressed.

The headband should be placed according to the user's best comfort, the device should be over the head, attached and adjustable by Velcro. Figure 3.2 shows the device in its front orientation.

Figure 3.2 - The final mounted set to smart headband on frontal face.

On the left side of the boxes there is a switch to turn on and off each device as well as a USB port to charge the Battery. On the front side there is another USB port for connecting the devices (Arduino Nano) to the computer. On the upper face (cover) there is a hole to view the status LED of the included Bluetooth module. This red LED should blink continuously if everything is working properly and the battery is charged. If the device does not have a battery or a connection is incorrect, the LED stops blinking. When the Bluetooth connection is established with the device, the status LED on the Bluetooth module stops blinking continuously and starts to do so at intervals.

Pairing the wearable devices with the computer via Bluetooth

To access wearable devices via the Bluetooth protocol, the computer where the system is installed must have this communication protocol on its source system, or it can be purchased in the market (e.g. USB Bluetooth Modules) and installed.

The pairing is performed in a single process which is described below (this procedure was performed on a computer running Windows 10 Home Edition):

1. Access the "System Taskbar" in the lower left corner of the "Action Center" on the Bluetooth option, with a right-click "Access Settings".

2. On the "Bluetooth and other devices" tab, you must enable the "On / Off" option to "On" and then go to the "Add Bluetooth or other device" button.

3. Then select the Bluetooth option. The name of each device will be displayed when connected. The devices have the following names:
   - **HC-05-LEFT,**
   - **HC-05-RIGHT,**
   - **HC-05-HEAD.**

4. Then choose the device you want to pair.

5. In the pairing security code for all devices, when set up, the value "**1234**" has been set and must be entered when prompted.

If this entire process has been successful, device names should appear on the page in the "Other Devices" list.

<u>Magnetometer Calibration</u>

Whenever the devices are to be used for the first time in a certain place it is necessary to calibrate the magnetometers present in the IMUs of the devices. This process is fundamental to acquiring trust values in the fusion algorithm to get good rotation readings. This process is very technical and implies that the user has some computer skills and because it is a prototype and is described below.

1. In order to calibrate the magnetometers, it is necessary to have access to the Arduino IDE computing platform and connect each of the devices with the computer via a USB cable to the front door of each device (Arduino Nano Port).

2. There is a folder in the Project Folder with all the content to apply on the Arduinos as well as the respective libraries to access the IMUs. Within this folder "*RTIMULib-Arduino-master"* there are other folders with the corresponding scripts for each of the devices, they are:
   - *"ArduinoIMU_head",*
   - *"ArduinoIMU_left",*
   - *"ArduinoIMU_right",*

however, the magnetometer calibration process is included in a different script in a different folder, which should be used on each of the devices:

- *"ArduinoMagCal"*.

3. Assuming the installation of the respective libraries present in the libraries folder and after opening the file, you must verify the type of board to use as well as the COM port, accessing through the Arduino IDE platform to the Tools> Board and Tools> COM port (see Figure 3.3).

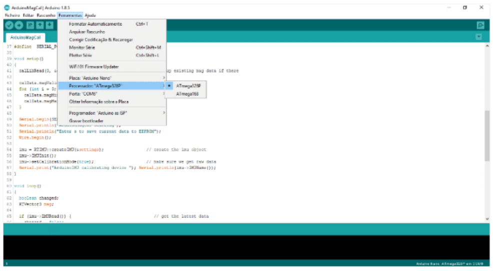

Figure 3.3 - Arduino IDE computation platform.

4. In the case of boards present on the hand devices the boards to be selected must be:

- Board: "Arduino Nano",
- Processor: "ATmega328P"
- COM port: "COM port number".

In the case of the board of the present in the device of the head the board to select should be:

- Board: "Arduino / Genuine One",
- COM port: "COM port number".

5. Then the script must be Uploaded for each of the boards.

6. If all has been successful, a Monitor Serial window should be opened to view and check the maximum and minimum values read by the magnetometer.

7. At this moment each device should be placed as close to where you want to perform the exercises and turn it in all directions. (This process can be facilitated if the cover of the box is removed to see the orientation of the IMU). When it no longer registers new highs and lows in the readings of the magnetometers an "s" command must be sent to the device to store the new ones in its internal memory (see Figure 3.4).

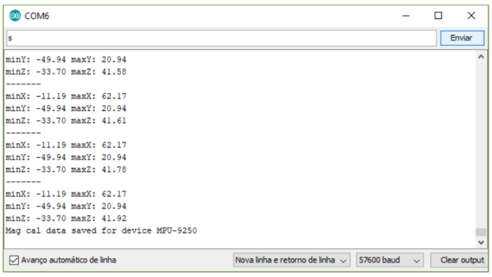

Figure 3.4 - Arduino IDE Serial Monitor.

8. After made the magnetometer calibration process is necessary replacing the data collection scripts in each of the devices. To do this, you must access the folder "*RTIMULib-Arduino-master"* again, open the respective folder of each of the devices ("*ArduinoIMU_head", "ArduinoIMU_left", "ArduinoIMU_right"*), follow the USB cable connection procedure mentioned above, and upload the file.

After this process, the wearable devices are ready to be connected to the system host computer via Bluetooth communication.

Appendix C – Technical Manual

DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY

Technical Manual

Physio Wear - Wearable and IoT Technologies Application for Physical Rehabilitation

Ricardo José Farinha Alexandre

Advisors:

Professor Octavian Adrian Postolache, Assistant Professor, ISCTE-IUL

Doutor Vítor Viegas, Assistant Professor, Portuguese Naval Academy

October 2018

Figures

Tables

Contents

Technical Manual

This manual aims to present and explain the different technical features of the PhysioWear project. Chapter 1 explains the Arduino class libraries and the developed scripts for each wearable device. Chapter 2 explains the classes and main functions of the PhysioWear Unity application. Chapter 3 explains how the server was configured to store Web API and Database. Chapter 4 is intended to explain the main functions of the mobile application

Chapter 1 - Embedded Devices

Overview

This chapter aims to explain the Arduino class libraries used as well as the developed scripts for each wearable device. The version used to develop the scripts was the Arduino IDE 1.8.5.

1. Arduino C scripts and libraries

In order to perform the data acquisition with the Arduino microcontroller, a set of scripts was developed, with support of open-source libraries. The microcontroller on the board is programmed with the Arduino programming language, based on the Wiring language, and the Arduino development environment, based on the Processing environment, that are essential for a correct use of the smart sensors.

The Arduino website provides a lot of good tutorials has several examples [1] to help the developer to learn and develop new applications. This same site also explains how to install the libraries in the system, and how to access them.

For our final prototypes of wearable devices, we considered the *RTIMULib-Arduino* library present in GitHub [2]. A more detailed explanation of the operation of the library is presented by the author and can be found in the latest version [3]. This library is responsible for collecting and performing all the primary processing of data collected by the IMU.

Our final scripts are based on the *ArduinoIMU.ino* script present in the library described above in the ArduinoIMU folder. The libraries folder contains all the library files that must be included in the system. At the beginning of each script can be found the #includes that make the accesses to these same libraries, which are presented next:

```
#include <Wire.h>
#include "I2Cdev.h"
#include "RTIMUSettings.h"
#include "RTIMU.h"
#include "RTFusionRTQF.h"
#include "CalLib.h"
#include <EEPROM.h>
#include <TimeLib.h>
```

In order to establish the Bluetooth communication, it is necessary to include a new library:

```
#include <SoftwareSerial.h>
SoftwareSerial BTserial(3, 2); // RX | TX
// Connect the HC-05 TX to the Arduino RX on pin 3.
// Connect the HC-05 RX to the Arduino TX on pin 2 through a voltage
divider,
```

This will establish the Arduino serial communication with the Bluetooth Module. At this point, the pins to be used in the connection must be defined. A good example of the Bluetooth Module configuration can be found in the tutorial described in [4]. Before this process, it is necessary access to the AT command list to properly configure the communication parameters. The access of the AT command list implies a change in electrical connection. A good tutorial to make the changes in connection as well as the AT command list can be found in [5].
Configuration of the AT commands was performed in Termite software, a simple RS232 terminal that allows interaction with the module.

For each of the wearable devices the following AT command settings per device were used:

Left hand:

- AT + NAME = HC-05-LEFT
- AT + PSWD = 1234 (Default)
- AT + ROLE = 0 (Slave Mode)
- AT + UART = 57600,0,0 (baud rate, stop bit, pairity bit)

Right hand:

- AT + NAME = HC-05-RIGHT
- AT + PSWD = 1234 (Default)
- AT + ROLE = 0 (Slave Mode)
- AT + UART = 57600,0,0 (baud rate, stop bit, pairity bit)

Head:

- AT + NAME = HC-05-HEAD
- AT + PSWD = 1234 (Default)
- AT + ROLE = 0 (Slave Mode)
- AT + UART = 57600,0,0 (baud rate, stop bit, pairity bit)

There is a folder in the Project Folder with all the content to apply on the Arduinos as well as the respective libraries to access the IMUs. Within this folder "*RTIMULib-*

Arduino-master" there are other folders with the corresponding scripts for each of the devices, they are:

- *"ArduinoIMU_head"*,
- *"ArduinoIMU_left"*,
- *"ArduinoIMU_right"*,

however, the magnetometer calibration process is included in a different script in a different folder, which should be used on each of the devices:

- *"ArduinoMagCal"*.

2. Programming

The **code snippet 1** taken from library [2] shows how obtain Euler Angles and to calculate Linear Acceleration using quaternion.

```
RTIMU *imu;                                                   // the IMU object
RTFusionRTQF fusion;  // the fusion readings (Euler Angles and Quaternions) object
RTQuaternion gravity, rotatedGravity; // quaternions to Gravity and RotatedGravity
RTQuaternion fusedConjugate;          // quaternions to Fused Conjugated readings
RTQuaternion qTemp;                 // quaternions to intermediate variable readings
RTVector3 realAccel;                // vector3 to save linear acceleration calculated
float gyro_x, gyro_y, gyro_z;                    // floats to save gyro readings
float accel_x, accel_y, accel_z;          // floats to save accelerometer readings
float mag_x, mag_y, mag_z;                 // floats to save magnetometer readings
float q_w, q_x, q_y, q_z;                        // floats to save quaternions readings
float roll, pitch, yaw;                 // floats to save euler angles readings

void setup(){…}
void loop(){
   // get the latest data if ready yet
   while (imu->IMURead()) {
      // this flush remaining data in case we are falling behind.
      if (++loopCount >= 10)
         continue;
      fusion.newIMUData(imu->getGyro(), imu->getAccel(), imu->getCompass(),
               imu->getTimestamp());
     // get gyro readings
     gyro_x = imu->getGyro().x() * RTMATH_RAD_TO_DEGREE;
     gyro_y = imu->getGyro().y() * RTMATH_RAD_TO_DEGREE;
     gyro_z = imu->getGyro().z() * RTMATH_RAD_TO_DEGREE;
     // get accelerometer readings
     accel_x = imu->getAccel().x() ;
     accel_y = imu->getAccel().y() ;
     accel_z = imu->getAccel().z() ;
     // get accelerometer readings
     mag_x = imu->getCompass().x() ;
     mag_y = imu->getCompass().y() ;
     mag_z = imu->getCompass().z() ;
     // get quaternion base readings
     q_w = fusion.getFusionQPose().scalar();
     q_x = fusion.getFusionQPose().x();
     q_y = fusion.getFusionQPose().y();
     q_z = fusion.getFusionQPose().z();
     // get euler angles readings
     roll = fusion.getFusionPose().x() * RTMATH_RAD_TO_DEGREE;
     pitch = fusion.getFusionPose().y() * RTMATH_RAD_TO_DEGREE;
     yaw = fusion.getFusionPose().z() * RTMATH_RAD_TO_DEGREE;
      // Linear acceleration calculation
      // 1 - gravity rotation and subtraction
      // 1.1 - create the conjugate of the pose
      fusedConjugate = fusion.getFusionQPose().conjugate();
      // 1.2 - now do the rotation - takes two steps with qTemp as the
      // intermediate variable
      qTemp = gravity * fusion.getFusionQPose();
      rotatedGravity = fusedConjugate * qTemp;
      // now adjust the measured accel and change the signs to make sense
      realAccel.setX(-(imu->getAccel().x() - rotatedGravity.x()));
      realAccel.setY(-(imu->getAccel().y() - rotatedGravity.y()));
      realAccel.setZ(-(imu->getAccel().z() - rotatedGravity.z()));
   }
}
```

Code Snippet 1 - how obtain the IMU associated data Euler Angles and Linear Acceleration using quaternions.

In order to be able to read 10 analog output sensors using the Arduino Nano platform, which only has 8 analog inputs, and 2 of them (SDA & SCL) are used by I2C technology for connection to the IMU module, a **Time Multiplexing** is performed on each analog input so that reliable readings of the two analog sensors associated with each finger can be obtained using a single analog input channel. In other words, for each clock cycle, two sensors are read on the same analogue input port.

The Arduino function *digitalWrite* (HIGH) command sets to 1 the digital input value that feeds the electrical circuit of the first analog sensor, then the *digitalWrite* (LOW) command sets this value to 0, a very short time delay (order of milliseconds) is applied and repeat the process for the second analog sensor. This procedure is performed simultaneously in each of the analog ports that will make the data acquisition of piezo-resistive force sensors. The **code snippet 2** show how it is done.

```
// For Each Analog Input Port
int analogPin0 = A0;                       // select the input Analog pin A0
int enable13 = 13;           // enable input Digital pin 13 to reading sensor A0_1
int enable12 = 12;           // enable input Digital pin 12 to reading sensor A0_2
int sensorValue1 = 0;       // variable to store the value coming from sensor A0_1
int sensorValue2 = 0;       // variable to store the value coming from sensor A0_2
int value1_min, value1_max;       // variables to store the values A0_1 to mapping
int value2_min, value2_max;          // variables to store the values A0_2 to mapping
int FlexiForce_A0_1;     // variable to store the value constrain from sensor A0_1
int flexSensor_A0_2;     // variable to store the value constrain from sensor A0_2

void getMultipleAnalogReadings(){
   // read the value from sensor 1:
   digitalWrite(enable13, HIGH);
   sensorValue1 = analogRead(analogPin0);
   FlexiForce_A0_1 = map(sensorValue1, value1_min, value1_max, 0, 100);
   digitalWrite(enable13, LOW);
   delay(2);                                                // delay milliseconds
   // read the value from sensor 2:
   digitalWrite(enable12, HIGH);
   sensorValue2 = analogRead(analogPin0);
   flexSensor_A0_2 = map(sensorValue2, value2_min, value2_max, 0, 90);
   digitalWrite(enable12, LOW);
   delay(2);                                                // delay milliseconds
}

void setup(){
   […]
   // For Each Analog Input Port
   pinMode(enable12, INPUT);
   pinMode(enable13, INPUT);
   pinMode(analogPin0, INPUT);
}

void loop(){
   […]
   // Get data of 2 analog sensors for each analog pin
   getMultipleAnalogReadings();
}
```

Code Snippet 2 - Shows how Time Multiplexing is performed to read 2 analog sensors on each analog pin.

Chapter 2 - Unity Application

Overview

This chapter aims to explain the main functions of the Physio Wear application for computer, in terms of programming. It also explains the description of the main classes used. The Unity version used to develop the application was the Unity 2017.3.1f1 (64-bit).

1. C# Classes

The application was developed according to the following diagram.

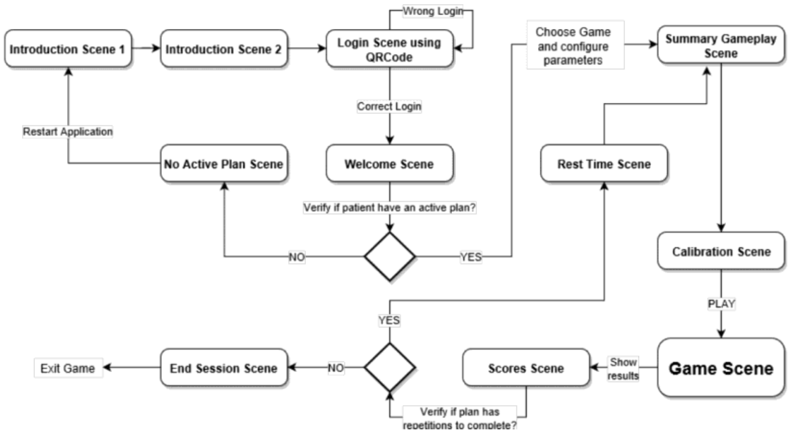

Figure 2.1 - Unity Application Diagram.

Following are the relevant scripts for the application, developed in C #, with a brief explanation and presented in alphabetical order. These scripts are present on Project Directory at Assets>Scripts folder.

Table 1 - Scripts used in Physio Wear Unity Application.

Audio_Manager.cs	This script is a utility used to load the audio clips to be used during the various game scenes.
CalibrationMode.cs	This script was developed to calibrate wearable devices before and during game scenarios. It is also responsible for scheduling the start of each exercise session.
Can.cs	This script was developed to act on the cans objects. Includes audio clips for interactions with cans.
Cans_Count.cs	This script is designed to count the can objects that fall to the ground during the game.
createSession.cs	This script was developed to register a workout in the database when a user initiates a new game.
Cups_Count.cs	This script is designed to count the cup objects that fall to the ground during the game.
debug_mode.cs	This script was developed to display some debug interfaces used during system development. This interface shows information about the rotation of the members and an estimate of the displacement of the members.
DirectorySaveCSV.cs	This script was developed to create a directory where to allocate the CSV files with the generated data of each game session.
EscapeQuit.cs	This script was developed to Close the game application by pressing the ESC key.
Exit_ApplicationGame.cs	This script was developed to exit the game application.
GameConfigurations.cs	This script was developed to configure all the parameters defined for the exercise session of the game scene.
Golden_cup.cs	This script was developed to apply a force on the ball objects that appear inside the golden cups.
HideMouse.cs	This script is a utility used to hide the mouse cursor during the entire application.
IMU_Head.cs	This script is designed to process all data received from the wearable device associated with the head. This includes connection establishment, calibration function, data acquisition using threads, the functions of acting on

	the head object, writing the data in CSV and sending the data to the database.
IMU_LeftHand.cs	This script is designed to process all data received from the wearable device associated with the left hand. This includes connection establishment, calibration function, data acquisition using threads, the functions of acting on the left hand object, movements of fingers, writing the data in CSV and sending the data to the database.
IMU_RightHand.cs	This script is designed to process all data received from the wearable device associated with the right hand. This includes connection establishment, calibration function, data acquisition using threads, the functions of acting on the right hand object, movements of fingers, writing the data in CSV and sending the data to the database.
Information_Loading.cs	This script was developed to load all the information related to each user's exercise plan.
MoveBallsByHand.cs	This script was developed to perform the implementation of the movements of grabbing and throwing the ball objects with the hands objects during Game Scene. It is also responsible for returning ObjectPool's ball objects whenever a ball is thrown.
MovePaddleByHand.cs	This script was developed to perform the implementation of the movements of grabbing and dropping the paddle objects with the hands objects during Game Scene. It is also responsible for returning ObjectPool's ball objects whenever a ball is thrown.
ObjectPool.cs	This script is a utility used to implement an ObjectPool that contemplates all objects that are generated during the Game.
PatientInformation.cs	This script was developed to store all the information related to a certain user, after logging in. The script is attached to an object that is passed in all game scenes, from the moment the Login is made. This script includes relevant data about the patient, as well as parameters about the exercise plan and scores.

PingPongBall.cs	This script was developed to act on the ping pong ball objects. Includes audio clips for interactions with table.
PingPongBall_Controller.cs	This script was developed to act on the ping pong ball objects. This script includes the ObjectPool generation functions.
PingPongBall_Count.cs	This script was developed to count the ping pong ball objects that fall to the ground during the game.
PingPongPaddle.cs	This script was developed to act on the ping pong paddle objects. It is also responsible for returning ObjectPool's paddle objects whenever a paddle is dropped.
QRCODE_AudioManage.cs	This script is a utility used to load the audio clips to be used during the Login game scene. Includes audio clips used on QR CODE Validation.
QRCODE_ScanEffect.cs	This script is a utility used to perform the Webcam scan effect in the Login scene.
QRCODE_Webcam.cs	This script is a utility used to perform the scan validation of the QR Code displayed on the webcam in the Login scene.
Restart_ApplicationGame.cs	This script was developed to restart the game application at the end of a set time.
RestTime.cs	This script was developed to count the Rest time of the exercises, when has repetitions according to the parameter configured in the exercise plan.
Rotator.cs	This script is a utility used to animate objects by applying a rotational force.
SaveScores.cs	This script is designed to perform the recording of punctuation data at the end of each exercise session.
SceneFadeInBlack.cs	This script is a utility used to perform the change of scenes of the game application using Fade In to a black screen.
SceneFadeInWhite.cs	This script is a utility used to perform the change of scenes of the game application using Fade In to a white screen.
Scores_ApplicationGame.cs	This script was developed to show score data at the end of each exercise session.

SerialPortSearcher.cs	This script was developed for the application to search the host system for the name of the corresponding COM port of each wearable device connected.
ShowPlanSummary.cs	This script was developed to show the Summary Plan, previously defined, at the beginning of each exercise session.
StartOnSpaceBar.cs	This script is a utility used to perform the scene change by pressing the SPACEBAR.
TennisBall.cs	This script was developed to act on the tennis ball objects. Includes audio clips for interactions with ground. It is also responsible for returning ObjectPool's ball objects whenever a ball is thrown.
TennisBall_Controller.cs	This script was developed to act on the tennis ball objects. This script includes the ObjectPool generation functions.
TennisBall_Count.cs	This script was developed to count the tennis balls objects that fall to the ground during the game.
TimerToEnd.cs	This script was developed to count the time of the Game Scene, according to the parameter configured in the exercise plan.
UI_ForceFeedback.cs	This script is designed to display pressure values read by FlexiForce sensors through coloured bars.
UnityMainThreadDispatcher.cs	This script is a utility used to allow Unity to run functions from other threads in the main thread.

In addition to these, there are in the Assets> Scripts> Models folder the scripts associated with Data Objects for the System Database.
The Project folder includes the Textures used present on Assets>Textures, the Materials used present on Assets>Materials, and the used Prefabs present on Assets>Prefabs.
The Final Scenes of Unity Application are present on Assets>Scenes;

Chapter 3 - Server Side

Overview

This chapter is intended to describe Physio Wear's Server Side and how communications are performed between the server and the other applications.

1. Database

The Database (DB) stores all of the data used and created by the applications, making it one of the main components of the Physio Wear system, and its design and implementation was done to ensure the correct operation of the system.
The entire database structure was developed in the MySQL Workbench v6.3 program [6]. This program has several tools to create a database through its visual presentation. The database was developed using MySQL v5.7.11. This Database Management System (DBMS) was chosen because it was the one that best suited the development needs of the project.
The database schema has been defined with the name:

- **my_wearable_iot_db**

however, all the data generated during the development of the system were allocated in a copy of the system, with the name:

- **my_wearable_iot_db_test1**

Security credentials are:

- **username: root**
- **password: root**

In Figure 3.1 it is possible to observe the final diagram of the database.

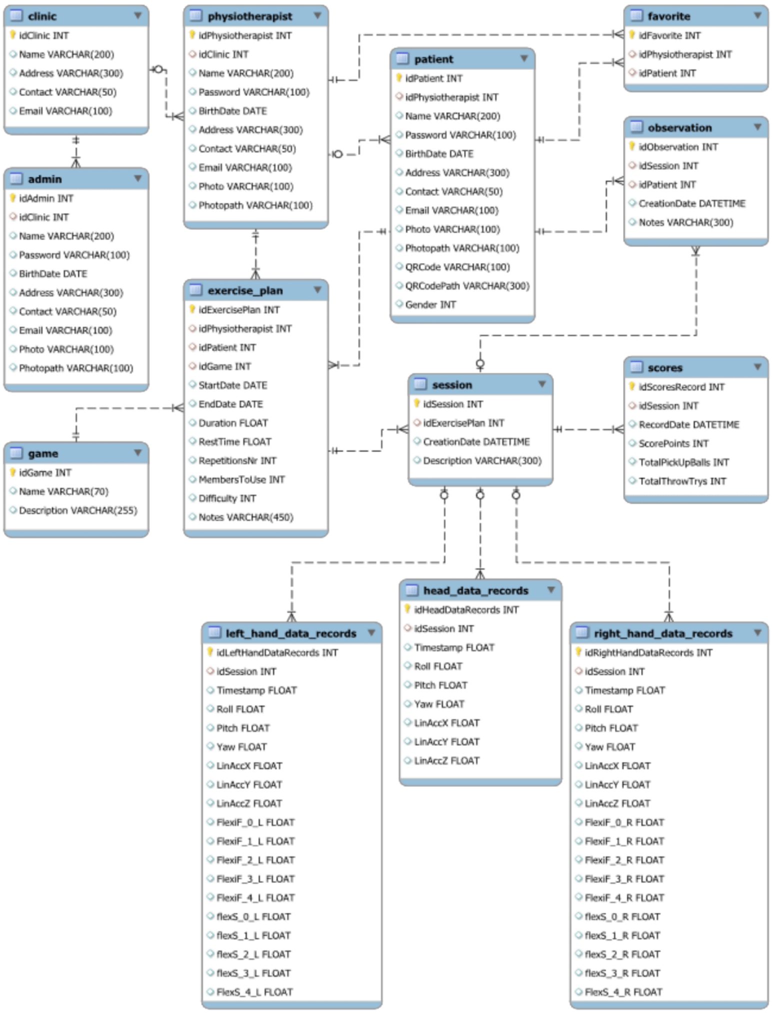

Figure 3.1 - Diagram Database.

The database consists of **thirteen** tables presented following:

Table 2 - The Tables of Physio Wear Database.

clinic:	This table stores the data related with the physiotherapy clinic. Fields such as the name of the clinic, address, contact and e-mail are present in this table. The primary key is idClinic.
admin:	This table stores the data for the clinic manager. It is in this table that the credentials (valid e-mail and password) are present for the manager to enter in the mobile application. Fields such as name, address and contact are present in this table. The table it is prepared to store the name of the manager's photo, through the Photo field, which is stored on the server. The Photopath field indicates the path to the folder where the manager's photo is hosted in relation to mobile app. Contains a foreign key (idClinic) that matches the clinical table, thus indicating which clinic the manager belongs to. The primary key is idAdmin.
physiotherapist:	This table stores the data concerning the physiotherapists of the clinic. It is in this table that the credentials (valid e-mail and password) are present for the physiotherapist to enter the mobile application. Fields such as name, address, date of birth and contact are present in this table. The table it is prepared to store the name of the physiotherapist's photograph, through the Photo field, which is stored on the server. The Photopath field indicates the path to the folder where the photograph is hosted in relation to mobile app. Contains a foreign key (idClinic) that matches the clinical table, thus indicating which clinic the physiotherapist belongs to. The primary key is idPhysiotherapist.
patient:	This table stores the data for patients in the clinic. It is in this table that the patient's credentials are present (valid

	e-mail and password) for them to enter in the mobile application. To access the Unity 3D App, they have a QRCode. This field is automatically generated by the mobile application and this is the content that is inserted in the QRCode field of the table, being an essential field to check the authentication. Fields such as the name, identification number, gender, address, contact, and e-mail are present in this table. The Photo and QRCode fields indicate the name of the photograph of each component that is stored on the server. The Photopath and QRCodePath fields indicate the path where the photograph is hosted in relation to mobile app. The gender field indicates the gender of patient. Contains a foreign key (idPhysiotherapist) that matches the physiotherapist table, indicating which physiotherapist is responsible for that patient. The primary key is idPatient.
favorite:	Each physiotherapist using the mobile application can mark a patient as a favorite. It is in this table that this information is stored. Contains two foreign keys, idPhysiotherapist and idPatient, which correspond to the physiotherapist and clinic table, respectively, thus indicating that a patient is signalled as a favorite for a physiotherapist. The primary key is idFavorite.
exercise_plan:	It is in this table that the configurations relative to each exercise planning are stored, being an essential table for the patients to execute the training created by the physiotherapists. The StartDate field indicates the start date for which this workout is available. The EndDate field indicates the end date of the workout, for example, The date by which the patient can perform the training. The Duration field indicates the time the patient must

	play each game. The RestTime field indicates the rest interval between two games. The RepetitionNr field indicates how many times that game must be repeated. The MembersToUse field indicates which hand to use (LEFT, RIGHT or BOTH). The Difficulty field indicates the difficulty of the game (EASY, MEDIUM or HARD). The Notes field allows the physiotherapist responsible for the training to write some observation. This table contains three foreign keys, the idPatient to indicate to which patient this training belongs, the idPhysiotherapist that indicates which physiotherapist is responsible for the training and the idGame that indicates the game to play. The primary key is idExercisePlan.
game:	This table stores the data regarding the games. It is in this table that this information is stored through field Name and Description. The primary key is IdGame.
observation:	This table stores the data regarding the observations created by the physiotherapists for the patients. This table has a field with the creation date of the observation and its description. Contains two foreign keys (idPatient and idSession) that matches the patient table, thus indicating which patient belongs to the observation and from what session realized. The primary key is idObservation.
session:	This table stores the data regarding the sessions performed by patients for the exercise plans created by physiotherapists. This table has a field with the creation date of the session and its description. Contains one foreign key (idExecisePlan) that matches the exercise_plan table, thus indicating which exercise plan that belongs the session completed. The primary key is idSession.

scores:	This table is intended to store the results of the training performed by each patient on the games. The RecordDate field stores the date that the result was entered the table. The ScorePoints, TotalPickUpBalls and TotalThrowTrys fields report to patient scores. It has a foreign key that matches the session table, thus indicating that certain results come from certain session, which exercise plan is performed by a patient. The primary key is idScoresRecord.
left_hand_data_records:	This table is intended to store the readings measured in the session for each patient when using the wearable devices, on this case for the left hand. The Timestamp field stores the time that the reading was entered in the table. The Roll, Pitch and Yaw fields store values about the rotations, the LinAccX, LinAccY and LinAccZ store values about the linear accelerations, all these measured by the IMU. The remaining fields are to store both the Flexi Force and Flex Sensor readings, all registered entries follow the order, from the little finger to the thumb. It contains a foreign key that matches the session table, thus indicating that certain measured readings come from certain session, which exercise plan is performed by a patient. The primary key is idLeftHandDataRecords.
right_hand_data_records:	This table is intended to store the readings measured in the session for each patient when using the wearable devices, on this case for the right hand. The Timestamp field stores the time that the reading was entered in the table. The Roll, Pitch and Yaw fields store values about the rotations, the LinAccX, LinAccY and LinAccZ store values about the linear accelerations, all these measured by the IMU. The remaining fields are to store both the

	Flexi Force and Flex Sensor readings, all registered entries follow the order, from the little finger to the thumb. It contains a foreign key that matches the session table, thus indicating that certain measured readings come from certain session, which exercise plan is performed by a patient. The primary key is idRightHandDataRecords.
head_data_records:	This table is intended to store the readings measured in the session for each patient when using the wearable devices, on this case for head. The Timestamp field stores the time that the reading was entered in the table. The Roll, Pitch and Yaw fields store values about the rotations, the LinAccX, LinAccY and LinAccZ store values about the linear accelerations, all these measured by the IMU. It contains a foreign key that matches the session table, thus indicating that certain measured readings come from certain session, which exercise plan is performed by a patient. The primary key is idHeadDataRecords.

The Photo and Photopath fields in the corresponding user tables of the system (admin, physiotherapist and patient) were not implemented in the mobile application due to time constraints and because they were non-fundamental components in the evaluation of the system. It is envisaged that they will be implemented in the future in order to make the profiles of users of the system even more interactive and reliable by presenting a photograph of the users.

2. Web API

The Physio Wear system uses a Web API to process the customer's request, access the DB and send the information, through C# scripts of a Web API created with ASP .NET Core v2.1 [7] using Entity Framework Core [8] to access a MySQL Database (version mysql-5.7.11).

Following is presented a table that includes all the Controllers created for access to the database models, as well as the request-response methods implemented in the Web API.

Table 3 - The Controllers created for access to the database models, as well as the request-response methods implemented in the Web API.

LoginController	
POST: api/Login:	This post method was created to check if the user's credentials to access the system (email address and password) are present in the database. It returns a negative response if it does not find the credentials, and positive if it finds them. The method performs the verification for the 3 types of system users (Admin, Physiotherapist and Patient).
ClinicController	
GET: api/Clinic/PerPatient/{idPatient}:	This get method was created to return a clinic based on idPatient. It returns a negative response if it does not find it, and positive if it finds it.
GET: api/Clinic/{idClinic}:	This get method was created to return a clinic based on idClinic. It returns a negative response if it does not find it, and positive if it finds it.
POST: api/Clinic:	This post method was developed to create a clinic in system. It returns a negative response if creation model is invalid and positive when successful.
PUT: api/Clinic/{idClinic}:	This put method was developed to update the contents of a clinic in the system based on idClinic. It returns a negative response if model is invalid, or idClinic does not match, and positive when successfully changed.

DELETE: api/Clinic/{idClinic}:	This delete method was developed to remove a clinic from the system based on idClinic. It returns a negative response if it does not find it, and positive if it finds it and successfully removes it.
AdminController	
GET: api/Admin/{idAdmin}:	This get method was created to return the admin data based on idAdmin. It returns a negative response if it does not find it, and positive if it finds it.
POST: api/Admin:	This post method was developed to create an admin in the system. It returns a negative response if creation model is invalid and positive when successfully.
PUT: api/Admin/{idAdmin}:	This put method was developed to change the contents of an admin in the system based on idAdmin. It returns a negative response if model is invalid, or idAdmin does not match, and positive when successfully changed.
DELETE: api/Admin/{idAdmin}:	This delete method was developed to remove an admin from the system based on idAdmin. It returns a negative response if it does not find it, and positive if it finds it and successfully removes it.
PhysiotherapistController	
GET: api/Physiotherapist/ PerPatient/{idPatient}:	This get method was created to return a physiotherapist based on idPatient. It returns a negative response if it does not find it and positive if it finds it.
GET: api/Physiotherapist/ ForClinic/{idClinic}:	This get method was created to return a list of all physiotherapists of a clinic based on idClinic.
GET: api/Physiotherapist/ {idPhysiotherapist}:	This get method was created to return a physiotherapist based on idPhysiotherapist. It returns a negative response if it does not find it, and positive if it finds it.
POST: api/Physiotherapist:	This post method was developed to create a physiotherapist in system. It returns a negative response if

	creation model is invalid and positive when successfully created.
PUT: api/Physiotherapist/ {idPhysiotherapist}:	This put method was developed to update the contents of a physiotherapist in the system based on idPhysiotherapist. It returns a negative response if model is invalid, or idPhysiotherapist does not match and positive when successfully changed.
DELETE: api/Physiotherapist/ {idPhysiotherapist}:	This delete method was developed to remove a physiotherapist from system based on idPhysiotherapist. It returns a negative response if it does not find it, and positive if it finds it and successfully removed it.
PatientController	
GET: api/Patient/ ForPhysiotherapist/ {idPhysiotherapist}:	This get method was created to return a list of all patients of a physiotherapist based on idPhysiotherapist.
GET: api/Patient/{idPatient}:	This get method was created to return a patient based on idPatient. It returns a negative response if it does not find it, and positive if it finds it.
POST: api/Patient:	This post method was developed to create a patient in system. It returns a negative response if creation model is invalid and positive when successfully.
PUT: api/Patient/{idPatient}:	This put method was developed to update the contents of a patient in the system based on idPatient. It returns a negative response if model is invalid, or idPatient does not match and positive when successfully changed.
DELETE: api/Patient/{idPatient}:	This delete method was developed to remove a patient from system based on idPatient. It returns a negative response if it does not find it, and positive if it finds it and successfully removes it.

GameController	
GET: api/Game:	This get method was created to return a list of all games of the system.
FavoriteController	
GET: api/Favorite/ {idPatient}:	This get method was created to return if a Patient is favorite based on idPatient. It returns a negative response if it does not find it and positive if it finds it.
GET: api/Favorite/ ForPhysiotherapist/ {idPhysiotherapist}:	This get method was created to return a list of all favorite patients of a physiotherapist based on idPhysiotherapist.
POST: api/Favorite:	This post method was developed to create a favorite patient for physiotherapist in system. It returns a negative response if creation model is invalid and positive when successfully.
ExercisePlanController	
GET: api/ExercisePlan/ ForPatient/{idPatient}:	This get method was created to return a list of all exercise plans of a patient based on idPatient.
GET: api/ExercisePlan/ ValidsPerPatient/ {idPatient}:	This get method was created to return a list of valid exercise plans of a patient based on idPatient.
GET: api/ExercisePlan/ {idExercisePlan}:	This get method was created to return an exercise plan based on idExercisePlan. It returns a negative response if it does not find it and positive if it finds it.
PUT: api/ExercisePlan/ {idExercisePlan}:	This put method was developed to change the contents of an exercise plan in system based on idExercisePlan. It returns a negative response if model is invalid or idExercisePlan does not match and positive when successfully changed.
POST: api/ExercisePlan:	This post method was developed to create an exercise plan system. It returns a negative response if creation model is invalid and positive when successfully.

DELETE: api/ExercisePlan/ idExercisePlan}:	This delete method was developed to remove an exercise plan from system based on idExercisePlan. It returns a negative response if it does not find it and positive if it finds it and successfully removed.
SessionController	
GET: api/Session/ ForExercisePlan/ {idExercisePlan}:	This get method was created to return a list of all sessions based on idExercisePlan.
GET: api/Session/ {idSession}:	This get method was created to return a session plan based on idSession. It returns a negative response if it does not find it and positive if it finds it.
POST: api/Session:	This post method was developed to create an session in system. It returns a negative response if creation model is invalid and positive when successfully.
ScoresController	
GET: api/Scores/ {idSession}:	This get method was created to return a list of all scores based on idSession.
POST: api/ Scores:	This post method was developed to create scores in system. It returns a negative response if creation model is invalid and positive when successfully.
ObservationController	
GET: api/Observation/ ForSession/{idSession}:	This get method was created to return an observation based on idSession. It returns a negative response if it does not find it and positive if it finds it.
GET: api/Observation/ {idObservation}:	This get method was created to return an observation based on idObservation. It returns a negative response if it does not find it and positive if it finds it.
POST: api/Observation:	This post method was developed to create an observation in system. It returns a negative response if creation model is invalid and positive when successfully.
PUT: api/Observation/	This put method was developed to change the contents of an observation in system based on idObservation. It returns

{idObservation}:	a negative response if model is invalid or idObservation does not match and positive when successfully changed.
LeftHandDataRecordsController	
GET: api/ LeftHandDataRecords/ {idSession}:	This get method was created to return a list of all LeftHandDataRecords based on idSession. It returns a negative response if it does not find it and positive if it finds it.
POST: api/ LeftHandDataRecords:	This post method was developed to create LeftHandDataRecords in system. It returns a negative response if creation model is invalid and positive when successfully.
RightHandDataRecordsController	
GET: api/ RightHandDataRecords/ {idSession}:	This get method was created to return a list of all RightHandDataRecords based on idSession. It returns a negative response if it does not find it and positive if it finds it.
POST: api/ RightHandDataRecords:	This post method was developed to create RightHandDataRecords in system. It returns a negative response if creation model is invalid and positive when successfully.
HeadDataRecordsController	
GET: api/HeadDataRecords/ {idSession}:	This get method was created to return a list of all HeadDataRecords based on idSession. It returns a negative response if it does not find it and positive if it finds it.
POST: api/HeadDataRecords:	This post method was developed to create HeadDataRecords in system. It returns a negative response if creation model is invalid and positive when successfully.

Chapter 4 - Mobile Application

Overview

The application was developed using Microsoft Visual Studio Community 2017 software version 15.7.5. Visual Studio is the official IDE for Xamarin Forms applications development. For its development in terms of programming was used C#.

1. C# Classes

The application was developed according to the following pattern:
The Model-View-ViewModel Pattern (see figure 4.1). A further description of this pattern is described in the official documentation [9].

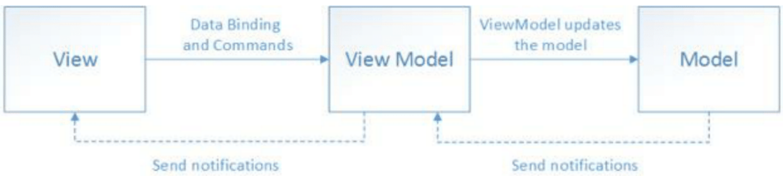

Figure 4.1 - The Model-View-ViewModel pattern.

View

The view is responsible for defining the structure, layout, and appearance of what the user sees on screen. Ideally, each view is defined in XAML, with a limited code-behind that does not contain business logic. However, in some cases, the code-behind might contain UI logic that implements visual behaviour that is difficult to express in XAML, such as animations.

In a *Xamarin.Forms* application, a view is typically a *Page*-derived or *ContentView*-derived class. However, views can also be represented by a data template, which specifies the UI elements to be used to visually represent an object when it's displayed. A data template as a view does not have any code-behind and is designed to bind to a specific view model type.

There are several options for executing code on the view model in response to interactions on the view, such as a button click or item selection. If a control supports commands, the control's Command property can be data-bound to an *ICommand* property on the view model. When the control's command is invoked, the code in the view model will be executed. In addition to commands, behaviors can be attached to an object in the view and can listen for either a command to be invoked or event to be raised. In response, the behaviour can then invoke an *ICommand* on the view model or a method on the view model.

ViewModel

The view model implements properties and commands to which the view can data bind to and notifies the view of any state changes through change notification events. The properties and commands that the view model provides define the functionality to be offered by the UI, but the view determines how that functionality is to be displayed.

The view model is also responsible for coordinating the view's interactions with any model classes that are required. There's typically a one-to-many relationship between the view model and the model classes. The view model might choose to expose model classes directly to the view so that controls in the view can data bind directly to them. In this case, the model classes will need to be designed to support data binding and change notification events.

Each view model provides data from a model in a form that the view can easily consume. To accomplish this, the view model sometimes performs data conversion. Placing this data conversion in the view model is a good idea because it provides properties that the view can bind to. For example, the view model might combine the values of two properties to make it easier for display by the view.

In order for the view model to participate in two-way data binding with the view, its properties must raise the *PropertyChanged* event. View models satisfy this requirement by implementing the *INotifyPropertyChanged* interface and raising the *PropertyChanged* event when a property is changed.

For collections, the view-friendly *ObservableCollection<T>* is provided. This collection implements collection changed notification, relieving the developer from having to implement the *INotifyCollectionChanged* interface on collections.

Model

Model classes are non-visual classes that encapsulate the app's data. Therefore, the model can be thought of as representing the app's domain model, which usually includes a data model along with business and validation logic. Examples of model objects include data transfer objects (DTOs), Plain Old CLR Objects (POCOs), and generated entity and proxy objects.

Model classes are typically used in conjunction with services or repositories that encapsulate data access and caching.

The Physio Wear project solution are organized like as following:

- PhysioWear.Models - accessed by mobile application and by Web API;
- PhysioWearMobile - It encompasses all the development classes that are shared by the different platforms. It includes the Application class (App.xaml), a script with constant values for database connections (Constants.cs), a folder with all application Views (pages), a folder with all application ViewModels, a folder with ApiServices.cs class (includes all WebAPI request-response methods), a folder with Helpers class (Settings.cs), a folder with values Converters, a folder with another generic Models.
- PhysioWear.Andoid - It encompasses all the development resources to Xamarin.Android mobile app, like UIs, Activities and drawable resources.
- PhysioWear.iOS - It encompasses all the development resources to Xamarin.iOS mobile app, like UIs, Configuration files and drawable resources.
- PhysioWear.UWP - It encompasses all the development resources to Xamarin.UWP mobile app, like UIs, Assets and drawable resources.
- PhysioWearWebApi - includes all WebAPI development classes, which includes all Controllers as well as access to Database Models.

2. Application Main Features

The mobile app was designed to be used by three types of users:

- The **patient:** to view his QR Code of access to the game platform, to consult plans of exercises and to see results of the sessions;

- The **physiotherapist:** to perform queries on patient data, create exercise plans, register patients or visualize results of sessions.
- The **clinic manager:** to register physiotherapists in the system or view information about them.

Below is a description of the features of each of the Mobile Application Pages (Views), present on PhysioWear.Mobile, at the Views folder and the respective ViewModels are present at ViewModels folder.

Table 4 – Mobile App Pages' description.

Splash Page	This page was developed as an entry page in the application, presents exclusively a logo animation.
Login Page	This page was developed to fulfil the application Login requests of the users of the application. After a registered user, enter their access credentials (valid e-mail address and PIN) can click on the Login button to make the request. There is a registration button that will serve as access to the clinic registration page and its administrator. And, a button to access the About page.
RegisterClinic Page	This page was developed to record new clinics in the system. This page includes a form that when completed is submitted to the registration request made after clicking the register clinic button. Then the registration page of the respective Administrator is loaded.
RegisterAdmin Page	This page was developed to make the records of the clinic administrators created in the system. This page includes a form that when completed is submitted to the registration request made after clicking the register clinic button. The page does not allow to go back, so it is necessary to fill the entire form. There will only be one administrator per clinic. After registration is done, the Login page is loaded again.
About Page	This page was developed to show information about application development. It includes a button for accessing the official Xamarin web page.

Home Page	This page which belongs to the Master Menu Page was developed to display a list of user profiles. In the case of the admin you can see the list of physiotherapists registered in the clinic. The physiotherapist will be able to view your list of patients. In the case of the patient the list is presented with the set of exercise plans registered for this patient. This page includes a search bar and an Add Profiles Page access button.
MyProfile Page	This page that belongs to the Master Menu Page was developed to show information about the users of the application. There is a button to access the information editing page. In the case of patients, there is another button to display the QR Code associated with the account profile.
QRCodeResultView	This view was developed to exclusively display the QR Code generated by the patient in your MyProfile Page after pressing the "See QR Code" button. This view is included in MyProfile Page.
Edit MyProfile Page	This page was designed to edit information about the users of the application. There is a form that will appear filled with profile information. At the bottom of the page there is a button that when pressed checks to see if there were any changes to register and makes the change request.
ViewClinic Information Page	This page that belongs to the Master Menu Page was developed exclusively to show the information of the clinic where the users belong. In the case of the patient, the information about your responsible physiotherapist will also be displayed.
AddProfile Page	This page that belongs to the Master Menu Page was developed to allow registration of profiles in the system. In the case of the administrators will be launched a form to fill a profile of physiotherapist and in the case of physiotherapists a form to fill the profile of patient. The patient does not have access to the contents of this page. Once the form has been filled out, there is a button that performs the request.

ViewProfiles Page	This page was developed to show information about the users of the application associated to parent profile. In the case of the administrators, the physiotherapists, and in case of physiotherapists, the patients. The patients do not have access to this page. In this page the physiotherapists can marker their favourite patients or delete it. There is a button to access the information editing page. In the case of physiotherapists, there is another button to display the exercise plans associated with the selected account profile.
EditProfiles Page	This page was designed to edit information about the users of the application associated to parent profile. There is a form that will appear filled with profile information. At the bottom of the page there is a button that when pressed checks to see if there were any changes to register and makes the change request.
View ExercisePlans Page	This page was developed to show the exercise plans list of the selected users of the application associated to parent profile. The patients do have access to this list on their Home Page. There is a button to access the information editing page. In the case of physiotherapists, there is another button to access the page to create new the exercise plans associated with the selected account profile.
Add ExercisePan Page	This page was developed to allow registration of exercise plans to patients in the system. A form to fill the exercise plan configuration is showed. The patient does not have access to the contents of this page. Once the form has been filled out, there is a button that performs the request.
ExercisePlan Page	This page was developed to show the information of the exercise plan selected on list. The page includes a list of all the sessions performed when performed. There is a button to access the information editing page.
Edit ExercisePlan Page	This page was designed to edit information about the Exercise Plan selected on list. There is a form that will appear filled with

	exercise plan information. At the bottom of the page there is a button that when pressed checks to see if there were any changes to register and makes the change request.
Report Page	This page was developed to show the information about the session selected from the performed sessions list. This includes the scores of the session and all charts of data collected from wearable devices during the exercise performance. Exists a set of buttons to controls the chart' views of each device. At the top there is a button to access the Observation page.
Observation Page	This page was developed to physiotherapist write some notes about the exercise performance described on Report Page. At the bottom of the page there is a button that when pressed checks to see if there were any changes to register and makes the change request. The patient can access this page only to read the contents.

Figure 4.2 illustrates the use cases that the users can perform in the application.

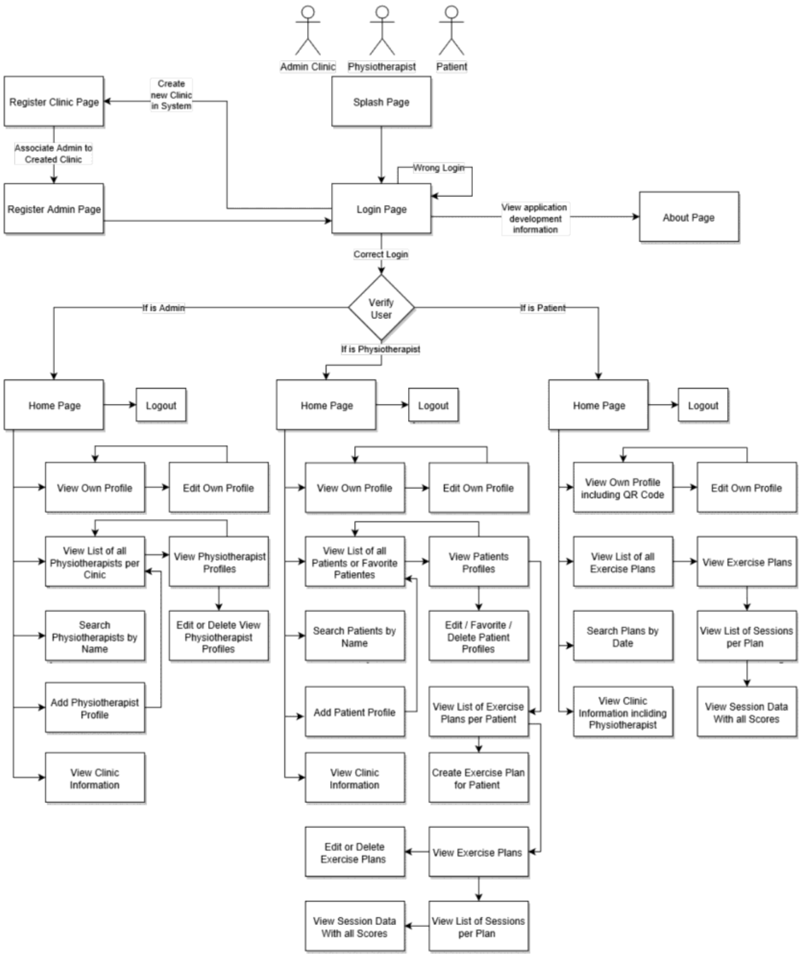

Figure 4.2 - Structure of the functionalities of the users in the mobile application.

Figure 4.3 illustrates the sequential diagram of the Pages in the application.

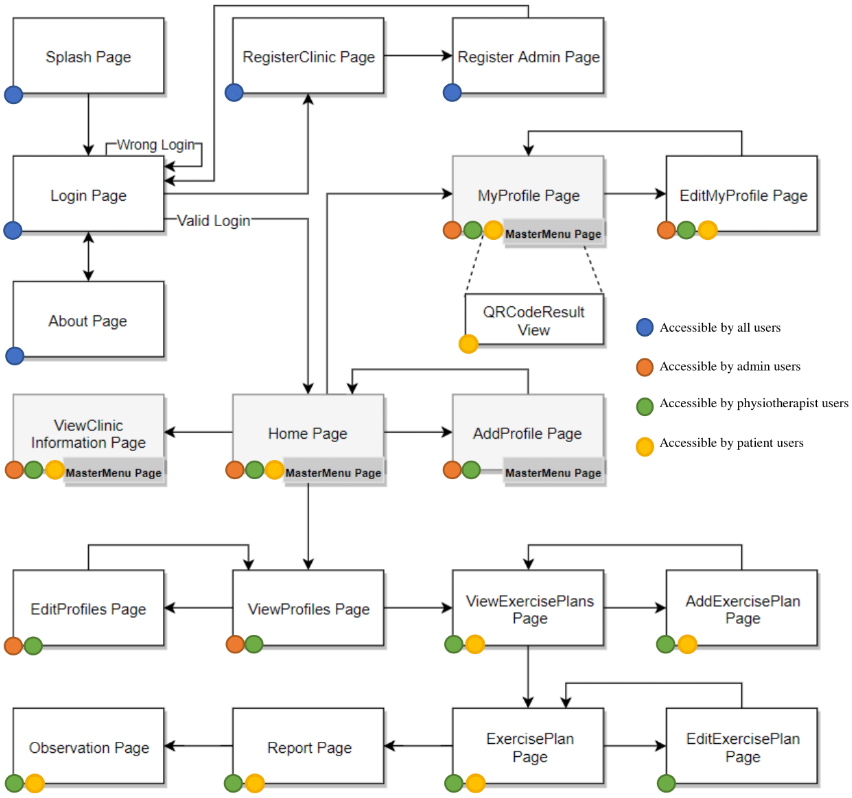

Figure 4.3 - Sequential Diagram of Mobile App.

References

[1] https://playground.arduino.cc/. (2018). *Arduino Library List*. [online] Available at: https://playground.arduino.cc/Main/LibraryList [Accessed 29 Sep. 2018].

[2] GitHub. (2018). richardstechnotes/RTIMULib-Arduino. [online] Available at: https://github.com/richardstechnotes/RTIMULib-Arduino [Accessed 2 Sep. 2018].

[3] GitHub. (2018). richardstechnotes/RTIMULib2. [online] Available at: https://github.com/richardstechnotes/RTIMULib2 [Accessed 29 Sep. 2018].

[4] Electrofun. (2018). Módulo Bluetooth HC-05 para Arduino. [online] Available at: https://www.electrofun.pt/comunicacao/modulo-bluetooth-arduino-hc05 [Accessed 19 May 2018].

[5] Instructables.com. (2018). How to Set AT Command Mode for HC-05 Bluetooth Module. [online] Available at: https://www.instructables.com/id/How-to-Set-AT-Command-Mode-for-HC-05-Bluetooth-Mod/ [Accessed 29 Oct. 2018].

[6] Mysql.com. (2018). MySQL :: MySQL Workbench. [online] Available at: https://www.mysql.com/products/workbench/ [Accessed 30 Jul. 2018].

[7] Docs.microsoft.com. (2018). Introduction to ASP.NET Core. [online] Available at: https://docs.microsoft.com/en-us/aspnet/core/index?view=aspnetcore-2.1 [Accessed 30 Jul. 2018].

[8] Docs.microsoft.com. (2018). Overview - EF Core. [online] Available at: https://docs.microsoft.com/en-us/ef/core/ [Accessed 30 May 2018].

[9] Docs.microsoft.com. (2018). The Model-View-ViewModel Pattern - Xamarin. [online] Available at: https://docs.microsoft.com/en-us/xamarin/xamarin-forms/enterprise-application-patterns/mvvm [Accessed 30 May 2018].